Face Page (Form DOE 4650.2)

DEVELOPMENT OF A MODULAR, PERFORMANCE-PORTABLE CLIMATE SYSTEM MODEL

For the period June 1, 2000, to September 31, 2001

Submitted to DOE Office of Science
March 24, 1999

Principal Investigator

Ian Foster
Associate Division Director, Mathematics and Computer Science, Argonne National Laboratory
Professor, Department of Computer Science, The University of Chicago

Argonne National Laboratory
9700 South Cass Avenue
Building 221
Argonne, IL 60439
(630) 252-4619 (voice)
(630) 252-5986 (fax)
foster@mcs.anl.gov

Co-Investigators, DOE Laboratories

Chris Ding	Lawrence Berkeley National Laboratory	chqding@lbl.gov
John Drake	Oak Ridge National Laboratory	bbd@msr.epm.ornl.gov
Phil Jones	Los Alamos National Laboratory	pwjones@lanl.gov
Jay Larson	Argonne National Laboratory	larson@mcs.anl.gov
Doug Rotman	Lawrence Livermore National Laboratory	rotman1@llnl.gov

Co-Investigators, NCAR

Thomas Bettge	Climate and Global Dynamics Division	bettge@ucar.edu
Maurice Blackmon	Climate and Global Dynamics Division	blackmon@ucar.edu
Byron Boville	Climate and Global Dynamics Division	boville@ucar.edu
Cecelia Deluca	Scientific Computing Division	cdeluca@ucar.edu
David Williamson	Climate and Global Dynamics Division	wmson@ucar.edu

Contents

1 Introduction ... 4
2 NCAR's Climate System Model ... 5
2.1 Overview ... 5
2.2 Atmosphere Model ... 7
2.3 Ocean Model ... 8
2.4 Land Surface, Sea Ice, and River Transport Model ... 10
2.5 Coupler ... 10
2.5.1 Coupling Design Issues ... 11
2.5.2 Coupler Performance Issues ... 11
2.5.3 The PCM-1 and CSM-1 Couplers ... 13
3 Target Platforms, Model Configurations, Software Engineering ... 13
3.1 Target Model Configurations and Throughput Goals ... 14
3.2 Target Platforms ... 14
3.3 Specific Parallel Performance Issues ... 16
4 Proposed R&D Activities ... 16
4.1 High-Performance Atmosphere Model ... 17
4.1.1 Accelerated Development of the High-Performance Atmospheric Model ... 18
4.1.2 Eulerian–Spectral Dynamical Core ... 19
4.1.3 Semi-Lagrangian-Spectral Dynamical Core ... 20
4.1.4 Lin-Rood Dynamical Core ... 20
4.1.5 Parallel Physics: Data Decomposition and Code Rework ... 21
4.1.6 Atmosphere - Land Surface Interface ... 21
4.1.7 Coordination Plan for the Atmospheric Model ... 21
4.2 High-Performance Coupler ... 22
4.2.1 The Handshake Problem ... 22
4.2.2 Parallel Regridding Software ... 23
4.2.3 Implications for Model Structure ... 23
4.2.4 Planned Activities for Flux Coupler Optimization and Development ... 24
4.3 Improvements to Coarse-Resolution POP Simulations ... 25
4.4 Parallel I/O ... 25
5 Code Development Methodologies and Infrastructure ... 26
5.1 CSM Software Developer's Guide ... 27
5.2 Design Documents ... 27
5.3 Staged Software Engineering Development Cycle ... 27
5.4 Formal Technical Reviews ... 28
5.5 Developers' Repository ... 28
5.6 Communication mechanisms ... 28
6 Management and Coordination ... 29
7 Relationship to Other Activities ... 30
7.1 Data Management ... 30
7.2 Computing Technologies ... 30
8 Tasks and Milestones ... 30
9 Resources ... 32
9.1 DOE Laboratories ... 32
9.2 NCAR ... 33
A. The Scripps/NCAR/PNNL ACPI Pilot Project ... 34
a. Introduction ... 34
b. Overview of Proposed Program ... 34
c. Main Program Elements and Their Status ... 34
i. Element 1: Ocean Assimilation ... 35
ii. Element 2: Modeling Anthropogenic Climate Change ... 35
iii. Element 3: Regional Downscaling/Impacts ... 36
Bibliographic References ... 37

Executive Summary

The Climate System Model (CSM-1) and Parallel Climate Model (PCM-1) of the National Center for Atmospheric Research (NCAR) are two advanced climate models that have seen significant use. The community CSM-1 model links atmospheric, oceanic, biologic, cryogenic, and chemical components; it has been and continues to be used for a wide range of climate research. Developed with DOE support, PCM-1 couples similar models and, in addition, has been adapted to execute on scalable parallel computers, hence allowing long-duration simulations in support of DOE missions.

Recognizing the strengths of these two models, NCAR scientists are merging the CSM-1 and PCM-1 code bases to produce CSM-2, with the goal of achieving significant improvements in model performance. As they tackle this goal, NCAR staff face two significant challenges. First, CSM-1 was not designed to exploit the microprocessor-based scalable parallel-architecture computers that are currently being deployed at NSF and DOE centers. A consequence of this limitation is that performance has not increased substantially in the past five years. Second, both CSM and PCM model structures could be improved with a view to enabling “plug and play” substitution of important modules, such as dynamical solvers and physics packages. This latter improvement will both facilitate ongoing development of the new merged CSM-2 model and make it easier for scientists to experiment with improvements to individual components.

A group of DOE and NCAR scientists thus propose a joint R&D activity aimed at developing a next-generation modular, performance-portable CSM-2. This work is expected to produce two primary outcomes: a performance-enhanced CSM-2, better able to exploit microprocessor-based parallel computers, and a detailed design for current and future CSM versions that improves substantially over current practice in terms of modularity and portability. A substantial challenge in both areas will be to evolve software engineering practices without unduly disrupting CSM development or diverging from a common code base.

The proposed R&D activity will tackle the design and development of (1) a scalable, modular atmosphere model and (2) a next-generation coupler. In the atmosphere domain, work will focus on improving node performance, developing a more modular atmosphere model structure that permits the substitution of both dynamics and physics components, and developing the high-performance communication libraries required for good performance on scalable parallel computers. Work on the coupler will address issues of scalability and configurability. Work on scalability is important because CSM-1 will not scale beyond around 64 processors. Work on configurability is important because users want to be able to use CSM components in a wide variety of modes more easily than with the current coupler. The DOE/NCAR team will also work on improving ocean model, sea ice model, and I/O performance on parallel computers.

This proposal specifies concrete, realizable tasks and milestones for both DOE and NCAR participants. These activities support the scientific directions for CSM, as defined by the CSM scientific steering committee, and also the goals of DOE’s CCPP Program. Day-to-day activities will be coordinated by a small management group, which we envision working closely with the recently formed CSM Software Engineering working group. It is expected that participants in the proposed project will be active participants in that working group. We note that a significant auxiliary outcome of this project will be the development and successful validation of the techniques required to enable productive collaborative development by a multi-laboratory and multi-agency team.

The project is proposed as an 18-month activity. We believe that this is an appropriate timeframe in which to be tackling these challenging CSM design and implementation tasks: a shorter project could not make useful progress, while a more ambitious but longer project might not have the required immediate impact on CSM development. However, we emphasize that the task of enhancing CSM for modular development and scalable parallel execution is a substantial project and while much can be done within 18 months, the effort will certainly not be completed at that time.

The proposed R&D activities complement and/or leverage activities funded or proposed under other programs: in particular, DOE funding for an “ACPI Pilot Project” (see the Appendix), a joint NSF-NASA program that is developing the so-called Lin-Rood dynamical core, NASA-funded activities focused on enabling collaborative model development, ongoing NSF funding for CSM, and DOE funding for the development of scalable numerical solvers and component models.

1 Introduction

The NCAR Climate System Model (CSM) project started in January 1994 and led to the release of CSM version 1.0 (CSM-1), which coupled atmosphere, ocean, land surface, and ice model components, in 1996. Various aspects of the model are described in a special issue of the *Journal of Climate* (Climate Systems Model 1998). Concurrent with this activity, a group led by Warren Washington at NCAR developed the Parallel Climate Model (PCM) with the goal of enabling long-duration climate simulations on scalable parallel computers (Washington et al. 2000). A merging of these two model development activities is proposed, with the goal of producing a substantially new community climate model, CSM-2, within the next year.

The developers of any major scientific simulation system—and in particular the developers of a system as complex as CSM-2—face two major challenges: performance portability and extensibility.

Performance portability arises as a challenge because of a simultaneous increase in both performance requirements (e.g., to support long-duration, high-resolution climate simulators, and ensemble studies) and in the range of computer architectures in use today. In addition to traditional vector machines, codes such as CSM must be able to operate efficiently on microprocessor-based distributed-memory computers, shared-memory computers, and hybrid systems. On these systems, memory hierarchies are complex, and memory bandwidth issues dominate performance. Future systems can only be expected to become more challenging in this regard (Sterling et al. 1995). Historically, NCAR models have emphasized vector performance and modest parallelism. New approaches to model development are required if we are to construct models that achieve good performance on a range of platforms while remaining usable by the scientists who must develop and maintain them. Successful projects at the European Center for Medium-Range Weather Forecasting and at Argonne National Laboratory in collaboration with NCAR's Mesoscale and Microscale Meteorology Division (Michalakes 2000) suggest that this goal can be achieved: but significant challenges remain, particularly in the context of more complex coupled models.

Extensibility arises as a challenge because the community nature of CSM means that a large number of developers need to experiment with modifications and extensions to the core CSM framework. For example, scientists located at geographically distant sites may incorporate advanced submodels, alternative dynamical cores, new physical parameterizations, and climate processes such as clouds, chemical interactions, and surface and subsurface water transport. If these modifications and extensions are not easy to perform, the utility of the model as an experimental framework is significantly reduced, the productivity of the CSM model development is curtailed, and the long-term scientific viability of CSM is compromised. Yet while NCAR models are well engineered on the whole, they have not been designed with a view to extensibility, modularity, and collaborative development. (For example, there is no overall "CSM design document.") Again, experiences elsewhere in the scientific computing community suggest that modularity can be achieved, even in the challenging case of high-performance scientific codes (Armstrong et al. 1999; Modular Design 199?); but achieving this goal in the context of CSM will require significant effort and commitment.

These considerations suggest that the utility of the CSM effort can be enhanced significantly by a focused attack on these two challenges. Furthermore, the fact that the CSM and PCM development teams at NCAR is at this moment engaged in development of CSM-2 means that such an activity would be particularly timely, making it possible to produce a CSM-2 that is significantly enhanced in these two areas. To this end, we propose here an R&D project with two main goals: first, to restructure key CSM components with a view to enhancing performance on a range of platforms, including scalable microprocessor-based parallel computers; and second, to develop the design principles and documents that can serve to guide future CSM development with a view to enhancing performance and extensibility. This project will be undertaken as a partnership between DOE laboratory scientists and software engineers, who provide expertise in parallel computing and software engineering, and NCAR scientists and software engineers, who provide expertise in computational physics as well as parallel computing and software development.

The rest of this proposal discusses both the technical content of the proposed work and various organizational issues that we believe must be addressed for the project to succeed.

Section 2 describes CSM and PCM, their principal components, and the performance characteristics of those components in their current form. The information in this section provides background information that informs subsequent technical discussion and also makes clear the nature of the performance challenges facing the developers of CSM-2.

The goal of performance portability implies that we wish to produce a model that can execute efficiently on a range of platforms and in a variety of configurations. To focus our efforts, we propose a small number of target platforms and resolutions, selected with a view to both meeting immediate NCAR and DOE needs and to covering the space of interesting configurations. These target platforms and resolutions are described in Section 3.

Section 4 describes the technical work proposed for this project. We focus on two primary CSM-2 components: the atmosphere model and coupler. Work on these two components, plus supporting work on performance tuning, parallel I/O, and ocean model performance, will produce a performance-enhanced CSM-2 able to achieve higher performance than the current CSM via both more efficient use of individual processors and efficient execution on larger numbers of processors. In addition, we anticipate as an important side product of this work that we will stimulate and contribute to the development of a detailed model design for CSM-2 as a whole. This model design will elucidate primary model components and interfaces and identify coding standards and testing procedures designed to enhance extensibility.

A particular challenge that we face in designing and executing this project is that we must engage CSM-2 developers in an “open software design processes” while simultaneously maintaining the productivity of NCAR scientists testing and using CSM-2. Only a well-planned and effectively executed project will build a long-term, sustainable model development collaboration and ensure productive joint work by multiple laboratories and multiple agencies. Hence, Sections 5 and 6 describes the design processes and management structure, respectively, that we will adopt to facilitate collaborative model development.

Section 7 discusses how the proposed activities relate to a number of other activities within NCAR, NASA, and DOE concerned with CSM development and software engineering

Section 8 describes the milestones that we expect to achieve in the project.

Finally, Section 9 describes how we propose DOE resources be allocated among DOE laboratories as well as the resources required at NCAR to support the proposed work.

An Appendix describes a companion project funded by DOE, an “ACPI Pilot Project” that will, we hope, serve as consumers for models developed in the project described here.

2 NCAR’s Climate System Model

We review the current and expected future evolution of the NCAR Climate System Model, focusing in particular on performance issues.

2.1 Overview

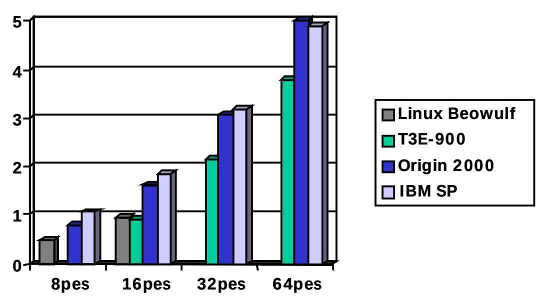
The effort to develop and support a community model for climate studies began 20 years ago. The first community atmospheric model, CCM0A, was described by Washington in 1982. This model was followed by CCM0B described in Williamson 1983. The second-generation community model, CCM-1, was introduced in 1987, and included a number of significant changes to the model formulation, which were manifested in changes to the simulated climate. The third generation of the CCM, CCM-2, was released in 1992 and improved the physical representation of key climate processes, including clouds and radiation moist convection, the planetary boundary layer and large-scale transport. The introduction of this model also marked a new philosophy with respect to implementation. The CCM-2 code was entirely restructured so as to satisfy three major objectives: greater ease of use, including portability across a range of computational platforms; conformance to a plug-compatible physics interface standard; and the incorporation of single-job multitasking capabilities. A steady improvement in the simulated climate of the CCMs is well documented along with more extensive treatment of physical processes.

An atmospheric climate model alone is not suitable for long-range studies of climate and climate variability. An active ocean general circulation model must be coupled with the atmospheric model even to simulate the important climate variations. The ENSO is an example of a coupling between the ocean and atmosphere that occurs on the interannual scale. In the late 1980s, as a result of advances in the scientific understanding of these interactions and the increased power of computing, the first community coupled models were developed. The NCAR CSM and PCM models are two of the premier next generation efforts.

Coupled atmosphere and ocean general circulation models (GCMs) are now becoming commonly used for studies of the natural variability of the climate system and its response to changes in greenhouse gases and aerosol radiative forcings. The National Center for Atmospheric Research (NCAR) Climate System Model, version one (CSM-1), is a physical climate model, similar in nature to several other coupled models that have been used for climate studies (see Gates et al. 1996 and Kattenberg et al. 1996). The main new features in CSM-1 compared with other coupled climate models are the coupling strategy and new state-of-the-art parameterizations, especially in the ocean model.

CSM-1 contains an atmospheric GCM, an oceanic GCM, a land surface biophysics and basic soil hydrology model, and a sea-ice dynamics and thermodynamics model. These component models communicate through a driver program called the flux coupler, which controls the time coordination of the integration and calculates most of the fluxes at the interfaces between the model components. The philosophy has been adopted in the CSM that the most appropriate boundary conditions for the component models are the fluxes at the earth's surface. Those interfacial fluxes that depend directly on the state of more than one component model – for example, turbulent fluxes of latent and sensible heat – are computed within the flux coupler. No flux corrections in momentum, heat, or freshwater are applied. The flux coupler is also responsible for interpolating and averaging between the different grids of the component models while conserving local and integral properties. The surface atmospheric fields are interpolated to the finer grid of the ocean model and the fluxes are calculated on the ocean model grid. The fluxes are then averaged back onto the coarser atmospheric model grid. This interaction becomes increasingly important if the ocean model has much higher resolution than the atmospheric model, because the higher resolution information affects the local turbulent fluxes.

The flux coupler currently allows two separate coupling intervals between itself and the component models. The atmosphere, land, and sea-ice models communicate at the faster interval, usually one hour, and the ocean model communicates at the slower interval, usually one day. Instantaneous values of state variables and interfacial fluxes time averaged over the coupling interval are passed. Therefore, fluxes are computed from instantaneous state variables, and the time integrals of the fluxes applied in the different model components are the same.



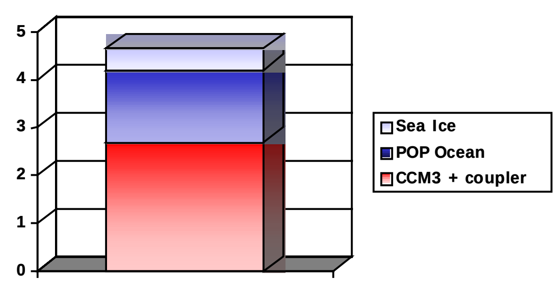


Figure 1: PCM performance in simulated years per wallclock day, on different computers (left) and wallclock hours per simulated year on a 64 PE SGI Origin 2000 (right) (T42 atmosphere, 2/3 ^{o} ocean).

The coupling strategy allows component models to be interchanged relatively easily. Each component model is isolated from the others and from the coupler, across a predefined message-passing interface. Therefore, different models can be used for any component without affecting the rest of the modeling system. For example, the ocean model can be a simple program to supply specified sea surface temperatures, or it can be the full ocean GCM. A tropical Pacific upper-ocean model can also be used for seasonal to interannual simulations. Similarly, the atmospheric component can be either CCM-3 or a program supplying results of previous simulations or atmospheric analyses. This flexibility is exploited during the spinup phase. The execution of the component models can even be distributed across different computers, a feature that has been demonstrated but is rarely used.

The Parallel Climate Model (PCM) is a coupled model with many of the same components as CSM-1. It has played an important role in DOE-sponsored simulations, and considerable effort has been invested to achieve good parallel performance (see Figure 1). The flux-coupling strategy of PCM differs from CSM-1 in the way the land surface model (LSM) is coupled with the atmosphere and in the manner in which component model execution is synchronized. The land surface and atmospheric coupling are built into the atmospheric component. The component models are configured as part of a single computer program, single executable, with components called sequentially. This strategy precludes the execution of component models across different computers; flexibility has been sacrificed in order to increase operational performance on target platforms.

The PCM and CSM ocean model components are based on different ocean models, Parallel Ocean Program (POP) for PCM and NCOM for CSM-1. The sea-ice models are also different with PCM using Zhang and Hibler (1997) sea ice dynamics with line relaxation for solving the viscous-plastic ice rheology (the rheology of Hunke and Dukowicz 1997 is also supported as an option) and CSM-1 using the rheology of Flato and Hibler (1992). Common component models are the atmospheric model CCM-3 and the land surface model (LSM). The merging of the PCM and CSM modeling efforts that will produce CSM-2 will require that significant scientific and software engineering design decisions be made. Plans are well advanced for CSM-2 to use versions of the POP and CICE codes for its ocean and sea-ice components.

The goals of this proposal support the established directions of NCAR scientists and CSM management. By seeking performance portability for CSM development, we will enable coupled climate simulations with adequate resolution to simulate weather systems, ocean eddies, and surface exchange processes that affect climate dynamics. By seeking an extensible design for CSM development, we will allow the incorporation of advanced submodels, parameterizations, and climate processes that will continue and accelerate the progress toward more comprehensive models that simulate climate with higher accuracy fidelity.

2.2 Atmosphere Model

The atmospheric GCM incorporated in both CSM and in PCM is CCM-3, which is described in Kiehl et al. (1998a, 1998b), Hack et al. (1998), Hurrell et al. (1998) and Briegleb and Bromwich (1998a, 1998b). CCM-3 is the latest generation of the Community Climate Model from NCAR with several major improvements over the previous versions (CCM-2), primarily in the parameterization of hydrologic processes and in the radiative properties of clouds. CCM-3 is a spectral model and the standard configuration, documented in the above papers, employs T42 truncation (~ 2.8 degrees) with 18 levels in the vertical. Penetrative convection is parameterized by the scheme of Zhang and McFarlane (1995), and the scheme of Hack is used for shallow convection. Cloud fractions and optical properties are computed diagnostically from large-scale variables and convective mass fluxes (Kiehl et al. 1998a). The nonlocal boundary layer turbulent flux parameterization is an updated version of Holtslag and Boville (1993), giving lower boundary layer depths and higher surface humidities. The long-wave radiation treats the effects of CO_2, O_3, H_2O, CH_4, N_2O, CFC11, and CFC12. With specified present-day sea surface temperatures, CCM-3 produces a globally and annually averaged balance between incoming solar radiation and outgoing long-wave radiation to less than 0.5 W/m^2.

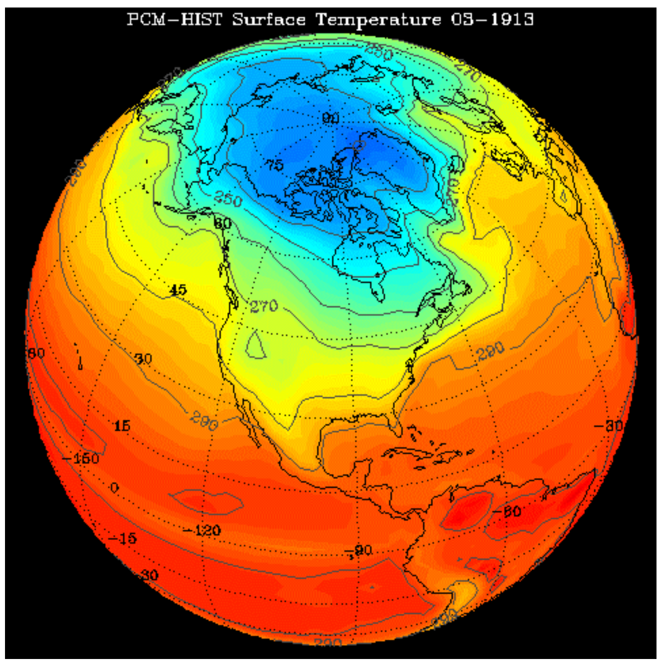


Figure 2: Surface temperature produced by CCM-3 in a PCM historical simulation

A 1D parallel decomposition is implemented in CCM-3, which provides for up to 64 parallel tasks in the grid point space and 43 parallel tasks in the spectral space at the standard T42 horizontal resolution. The decomposition by latitude has been used successfully on moderately sized systems, and for the past several years, most of the CSM and PCM simulations have been performed with configurations of 32 or 64 processors. Studies of performance based on a more general 2D parallel decomposition of CCM-2 (Drake et al. 1995) and CCM-3 indicate that the longitude direction should also be decomposed when using more than 32 processors. The optimal aspect ratio depends on machine characteristics; on the NERSC Cray T3E-900, 512 processors can best be utilized with a 16(lon)x32(lat) decomposition. For T42 resolution, parallel efficiency begins to decline after 32 processors, with performance per processor dropping by half at 512 processors. As an indication of what a more general parallel decomposition can yield on current platforms, Figure 3 shows the time per simulated day for the atmospheric model at T42L18. The study used the Cray T3E-900 at NERSC, the IBM SP3/WinterHawk1 and WinterHawk2, the Compaq AlphaSC-500 at ORNL, and the SGI Origin2K at LANL. The situation becomes even more complex with the reduced grid that has been introduced into CCM (Williamson and Rosinski 2000). This introduces a ragged array (lon,lat) rather than the rectangular array of the full grid. These performance results for a single model component at low resolutions suggest that further throughput gains can be made and underscores the need for careful design and implementation of future models.

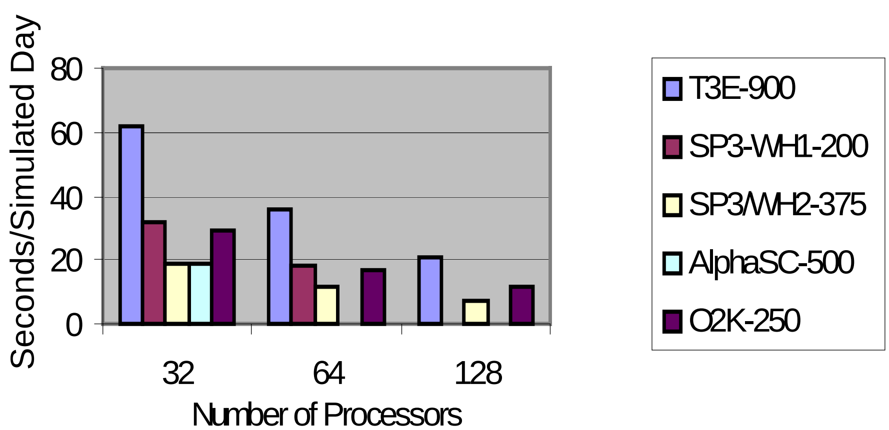
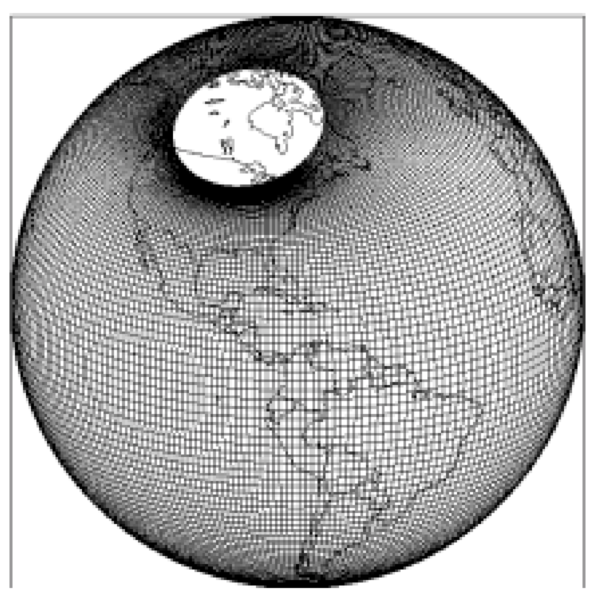


Figure 3: CCM-3 performance on various parallel computers when using a 2-D decomposition.

2.3 Ocean Model



The ocean component planned for CSM-2 (and used in the current PCM) is the Parallel Ocean Program (POP) from Los Alamos National Laboratory. POP is a descendant of the Bryan-Cox-Semtner (BCS) class of z-level ocean models and was developed under the sponsorship of the Department of Energy's CHAMMP program. A number of improvements were developed and incorporated in POP both for performance on parallel computers and for improved ocean simulations. Significant algorithmic improvements over previous BCS models include a surface pressure formulation and implicit solution of the barotropic mode, a free surface boundary condition and pressure averaging.

These improvements permit longer time steps and allow the use of unsmoothed, realistic topography. Details of the model are found in articles by Smith et al. (1992), Dukowicz et al. (1993), and Dukowicz and Smith (1994). POP also supports general orthogonal grids in the horizontal coordinates. In particular, the use of displaced-pole grids (see Figure 4, Smith et al. 1995) in which the pole in the northern hemisphere is displaced into land masses permits accurate solutions of the Arctic regions without filtering or severe time step restrictions related to convergence of grid lines. A number of improved physical parameterizations have recently been added to POP. These are the Gent-McWilliams (1990) isopycnal mixing scheme, the KPP vertical mixing scheme (Large et al. 1994), and an anisotropic horizontal viscosity. The last parameterization gives improved equatorial currents at coarse resolution, but is not necessary at higher resolutions, such as 2/3°.

POP was designed from the beginning to run on massively parallel computers and has continued to evolve as machine architectures have evolved. It is designed to be very portable and runs on most available machines with *no* changes in the source code and *no* architecture-specific preprocessor options. Interprocessor communication details are encapsulated in a very few modules that support MPI, Cray SHMEM, and serial execution; with the choice of communication paradigm being determined by specifying the appropriate directory during the build process. Work is currently under way on a hybrid MPI/OpenMP implementation that will allow more efficient use of clusters of SMP boxes.

Performance of POP on SGI Origin 2000 machines and on IBM SP (Winterhawk I) machines is shown in Figure 5. Node performance is typically around 10% of peak performance on cache-based microprocessors and around 50% of peak for vector machines. Node performance is largely limited by the extensive use of array syntax in POP and the inability of current Fortran compilers to efficiently fuse such array expressions into larger loops that can take advantage of cache reuse. Node performance has been improved by as much as 40% in an experimental MPI/OpenMP version of POP because the subblocks over which the OpenMP threads operate can be tuned efficiently to fit into secondary cache.

Scaling of POP on large numbers of processors is limited primarily by the implicit solution of the barotropic mode. The two-dimensional barotropic mode is solved by an iterative solver and involves the application of a nine-point stencil operator followed by global sums. At high processor counts or for coarse horizontal resolution, the solver has very little work to do on the processors and sends many very small messages for global sums and boundary updates. The barotropic solver is thus latency dominated and can exhibit poor scaling on machines with high-latency networks. Since century-scale CSM runs require ocean resolutions between 1 and 3 degrees, poor scaling of the barotropic solver will hinder the overall performance of coupled CSM simulations.

The difference between the total and baroclinic lines in Figure 5 is the barotropic solver; note the poor scalability in some situations. The 384x288x32 result corresponds to the 2/3° grid.

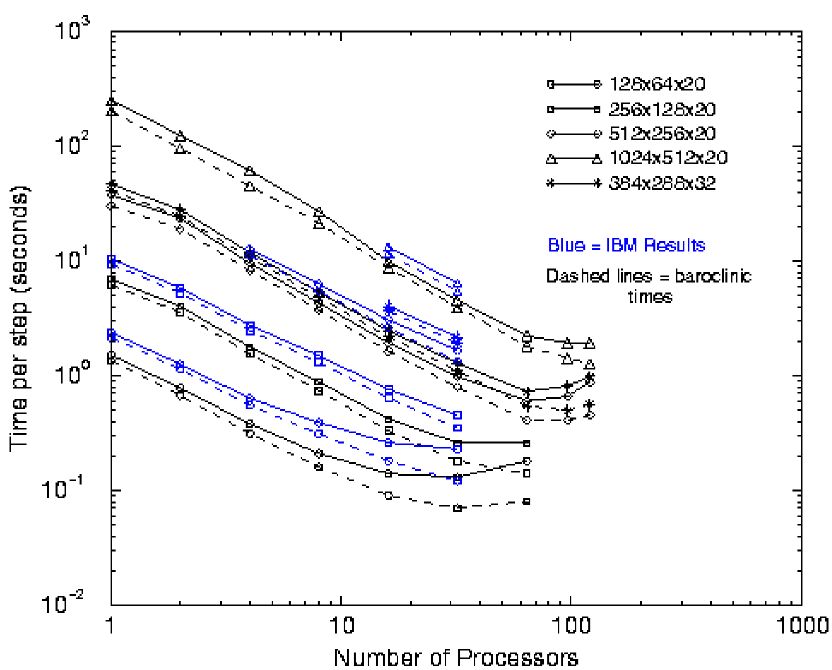


Figure 5: POP performance on SGI 02000 and IBM SP Winterhawk-1

2.4 Land Surface, Sea Ice, and River Transport Model

The NCAR Land Surface Model (Bonan 1998) simulates the biogeophysical and biogeochemical land-atmosphere interactions, especially the effects of land surfaces on climate and atmospheric chemistry. The LSM runs on the same grid as CCM-3, but rather than attempting to define an average land and vegetation type for each grid cell, the cells are subdivided into four different surfaces, allowing differing vegetation type, bare soil, lakes, and wetlands to be treated separately. Grid cell average fluxes are determined by area averaging the fluxes of each surface type. A river runoff model has recently been included that balances freshwater in the CSM. The LSM surface fluxes are tightly coupled with the atmospheric GCM but operate as a separate executable in the CSM. Currently, surface points containing land are statically partitioned to processors, with MPI-style communications flowing through the flux coupler.

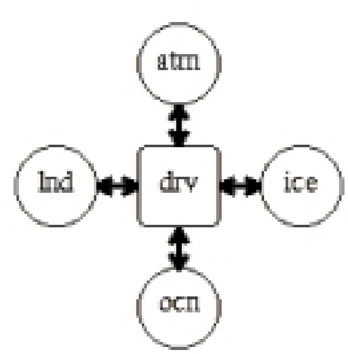
We do not have reliable scaling numbers for LSM, but believe that this component model scales well, at least for the processor configurations envisioned in this project. This belief is substantiated by the fact that in practice the LSM takes a small fraction of the overall CSM execution (i.e., wallclock) time for typical processor configurations. As part of this project, we plan to analyze LSM scalings and confirm these beliefs.

The sea-ice component of CSM-2 will be based on the CICE framework developed at Los Alamos National Laboratory by Hunke and Lipscomb (1999). This model contains three interacting components: (1) a thermodynamic model based on Bitz and Lipscomb (1999) produces local growth rates of snow and ice due to vertical conductive fluxes, snowfall and local temperatures; (2) a model of ice dynamics predicts the velocity field of the ice pack based on a model of the material strength of the ice; and (3) a transport model describes advection of the areal concentration, ice thickness, and snow depths. CICE uses an elastic-viscous-plastic rheology for improved ice dynamics and an improved ice advection algorithm. The model will incorporate multiple ice thickness categories, initially following the work of Bitz et al (2000). CICE is fully explicit in its time integration and is implemented using both MPI and OpenMP parallelism. Results show that the CICE model scales well for small numbers of processors (5 to 10 processors); but because sea ice is essentially a two-dimensional phenomenon at the ocean/atmosphere boundary, scaling may be limited at high processor counts where the surface-to-volume ratio is large. The scalability of the CICE model will be further investigated under this proposal.

2.5 Coupler

The coupler is the model component responsible for coordinating the execution of the various components that form a coupled mold. It serves three primary functions (see http://www.cgd.ucar.edu/csm/models/cp1/doc4 and http://www.cgd.ucar.edu/csm/models/cpl).

- It allows the model to be broken down into *separate components*, atmosphere, sea-ice, land, and ocean, that are "plugged into" the Flux Coupler ("drive"). Each component model is a separate code that is free to choose its own spatial resolution and time step. Individual components can be created, modified, or replaced without necessitating code changes in other components, unless new parameterizations require more information from the other components.
- It *controls the execution* and time evolution of the complete model by controlling the exchange of information among the various components. (It also provides rudimentary fault detection.)
- It *computes interfacial fluxes* among the various component models (based on state variables) and distributes these fluxes to all component models while insuring the conservation of fluxed quantities.
- It handles the *mapping* operations required to transform data between the different grids used in different components.



As illustrated in Figure 6, one way of thinking about the coupler is as a physically distinct process that arbitrates and implements all information flows among model components. This also turns out to be how the coupler is implemented in CSM-1 (but not PCM).

More abstractly, we can think of the coupler as a set of regridding, communication, and synchronization operations. This alternative view of coupling allows for more flexibility in implementation, as these various functions can be invoked wherever makes the most sense from a performance and software engineering viewpoint: within different model components or, alternatively, centralized in a distinct coupler process. This is the view that we will emphasize in this proposal.

2.5.1 Coupling Design Issues

A number of issues complicate the design and implementation of coupling functions, in particular, the following:

- *Sequencing.* Corresponding timesteps of the coupler and the various component models may be executed in sequence or at the same time (concurrently). In the sequential approach, each component always has access to the latest state information from other components. The concurrent approach permits parallel execution, which is essential if components are to be distributed. However, it requires that one set of fluxes (typically the atmosphere) must be lagged by one timestep to achieve parallelism between the component models. In addition, serial dependencies between components can result in idle processors.
- *Frequency of communication*. The various model components may exchange information at every time step or less frequently. There are obvious tradeoffs to be made between performance and accuracy.
- *Distribution.* The coupler and the various component models may be executed on the same processors ("stacked") or, alternatively, on disjoint sets of processors ("distributed"). The parallel approach can have performance advantages, as intercomponent communication tends to be less than intracomponent communication. On the other hand, the stacked approach avoids the need to compute efficient allocations of processors to components; if this is not done (or is not possible), then load imbalances occur. Note that hybrid approaches are possible: some components may be stacked (e.g., land and atmosphere in many models) while others are distributed.
- *Coupler parallelism.* Because different components use different grids and different representations of quantities of interest, the coupler can be required to perform considerable communication and computation. Hence, parallelism within the coupler can be important.
- *Separate or single executable.* The various model components can be linked into a single executable or (in a distributed approach) maintained as separate executables.
- *Support for standalone execution.* Individual model components will be used at different times in two different modes: as part of a coupled system or "standalone" with boundary conditions obtained from a file. A well-designed coupler can avoid the need for two different versions of each component, by allowing boundary conditions to be obtained via the same mechanisms in each case.

2.5.2 Coupler Performance Issues

The flux coupler controls the exchange of interfacial information between the component models of the coupled climate system. The primary and defining requirement of the flux coupler is that it must ensure that conservative properties, such as momentum, heat and fresh water are numerically conserved in the exchanges between component models. This requirement is complicated by the fact that different models use different types of grids at diverse resolutions. Thus the conservative regridding of interfacial fields between component grids is a key aspect of the flux coupler. As will be seen later, these regriddings present two serious software engineering problems. The first problem stems from the fact that the number of possible regriddings scales as the number of components squared. This "handshake" problem affects the complexity of the flux coupler in every respect. The second problem is derived from the fact that these regriddings are not especially well load balanced and not easily parallelized.

One of the factors limiting the performance of the current flux coupler is a load imbalance in the remapping of fields between component model grids. In a conservative remapping, grids cells on one grid must communicate with any grid cell on the other grid with which it overlaps. If both grids are simple latitude-longitude grids, then the communications pattern in the remapping is regular because both grids are regular. However, in the case of POP displaced-pole grids (see Figure 4), large communication imbalances occur for two reasons. First, POP grid cells can vary greatly in size; cells near the equator are much larger than cells near the poles of the grid. Second, the ocean grid cell that overlaps the North Pole covers all (128 for a T42 grid) the cells of the atmosphere grid that are converging at this pole point. Both of these result in cases where some grid cells need to communicate with a very large number of cells on the other grid while other grid cells only need to communicate with a few cells on the other grid. Such a disparity causes a large communication load imbalance.

In addition to regriddings, the flux coupler performs flux calculations. These calculations are typically performed on the finest grid in the coupled system for accuracy reasons. Thus a typical scenario for flux coupler operation is to receive a set of 2D state variables and input fluxes from a component on a particular grid. These fields are then regridded to the highest resolution grid in the climate system, on which output fluxes are calculated. The resulting output fluxes are interpolated back to the original component grid and returned to that component.

Another complicating factor when designing efficient coupling schemes is the considerable dynamic range in the number of regridding calculations required, depending on problem resolution. For example, compare the relative costs of a $2/3^o$ ocean to T42 atmosphere regridding with a 3^o to T42 atmosphere regridding. The regridding cost varies according to the number of grid points, that is, by a factor of $O((3/(2/3))^2) = \sim 20$ in this case. Other combinations (e.g., high-resolution atmospheres and statistical models) will likely result in coupler performance demands that are dramatically different in other dimensions.

The following numbers are from 10-day PCM tests on the IBM SP at NCAR and the Origin 2000-250MHz at LANL. Times are in seconds for coupler as a whole and for each mapping (e.g., a2o = atmosphere to ocean grid). Grids are T42 atmosphere, $2/3^o$ ocean, and 27 km ice. The total number of mappings for each category is shown in the following table:

a2o:	7200
o2a:	5910
i2o:	150
o2i:	180

The residual (total minus sum of mappings) is the total time spent in the other functions of the PCM coupler–conservation, summing, averaging, diagnostics—but gives us no specific information about any of them.

<u>IBM SP</u>

PEs	Total PCM Coupler Time	a2o	o2a	i2o	o2i	Mappings as a % of total PCM	Fcd as a % of total PCM
8	194	49	29	5	2.9	44%	8%
16	118	26	16	4.1	2.7	42%	9%
32	108	24	19	4.1	2.5	46%	14%
64	98	24	17	3.7	2.3	48%	18%

<u>Origin 2000</u>

PEs	Total PCM Coupler Time	a2o	o2a	i2o	o2i
8	177	32	17	3.9	2.3
16	98	19	9	2.6	1.8
32	69	15	10	2.2	1.4
64	67	18	14	1.6	1.4

Clearly, scaling is an issue with the mappings, but the mappings are not the only function limiting the scaling of the PCM-1 coupler. Conservation and diagnostics computations (global reductions) probably also contribute.

With respect to the mappings themselves, there are two issues: moving data (messaging passing latency and bandwidth issues) and load imbalances (sparse matrix multiply where work is not evenly distributed). Preliminary data suggest that message passing costs exceed computational costs for a modest number of processor elements; furthermore, message-passing costs increase with the number of processor elements.

2.5.3 The PCM-1 and CSM-1 Couplers

Previous work at NCAR has resulted in the development of two distinct coupling strategies. PCM-1 uses a sequential, stacked strategy, in which all model components execute in sequence on the same processors. CSM-1 uses a concurrent, distributed strategy, with ocean, atmosphere, ice, land surface, and coupler each executing simultaneously on disjoint processors. The CSM coupler uses the SCRIP remapping package from Los Alamos National Laboratory (http://www.acl.lanl.gov/lanlclimate/SCRIP), which implements the general remapping scheme of Jones (1999) for performing conservative remapping between any two grids on a sphere. In the case of surface fluxes, these remappings must be (and are) performed in a conservative manner.

Coupling with the ocean currently occurs once per model day. Atmosphere, land, and ice couple much more frequently: between once per hour and once per atmosphere timestep (20 minutes). In general, land/atm/ice interactions with the coupler are at least 1 order of magnitude more frequent that in the case of the ocean. The following table summarizes the coupling frequencies and volumes to be used in CSM-2.

Component	**Coupling frequency**	**Communication**	
		Component -> Coupler	**Coupler -> Component**
Land Surface Model	Once per hour	6 states, 6 fluxes	7 states, 9 fluxes
Ice Model: CICE	Once every two hours	6 states, 13 fluxes	11 states, 10 fluxes
Atmosphere model: CCM	Once per hour	7 states, 10 fluxes	6 states, 6 fluxes
Ocean model: POP	Once per day	4 states, 3 fluxes	6 fluxes

Experiences with PCM and CSM illustrate some of the tradeoffs noted in the list of coupling issues above. In PCM, for example, we find that the model scales quite well up to 64 processors (on IBM SP and SGI Origin) but that scaling drops off beyond that point (see Figure 1). This lack of scaling is a result of poor parallel efficiency within the more two-dimensional models used for the ice, land and flux coupler, which in the PCM strategy must execute on all processors.

With CSM-1, we encounter both load-balancing problems and scaling problems with the coupler. As an example of scaling problems, on 64 IBM Winterhawk-1 nodes, the atmosphere model at T42L18 takes around 30 seconds per model day (0.5 seconds per time step). In this configuration, the coupler running on a single processor takes 15 seconds per simulated day to execute the atmosphere critical path (the code that cannot be overlapped with atmosphere model execution) when coupling with a 3^o ocean. Hence, the coupler needs to be sped up by a factor of 5 to get the overhead down to about 10%, and about 30 to get the overhead down below a single T42L18 CCM timestep. Multithreading has been used to a limited extent within the CSM-1 coupler, but has not overcome this performance problem.

Thus, history has left us two very different flux coupler designs, one threaded and one pure message passing. At the moment, neither implementation is particularly well suited to modern parallel systems composed of clusters of multiprocessors, which require a hybrid parallel programming approach for optimal performance. The question then becomes, How can a more general design for the flux coupler be derived that is more scalable, more extensible, and less susceptible to changes in computer system design and capabilities?

3 Target Platforms, Model Configurations, Software Engineering

It is infeasible from a software maintenance point of view to have other than a single version of a model source code. Hence, an overriding goal of this project is to develop a *performance-portable* CSM: that is, a code that is able to execute efficiently on a variety of platforms and in a range of model configurations. It is nevertheless useful to identify initial model configurations and target platforms, with the goal of defining the space within which performance portability is required.

3.1 Target Model Configurations and Throughput Goals

The "standard" CSM configuration is currently 3^o (T42) atmosphere and 2^o ocean. DOE-sponsored simulations designed to provide input for regional climate studies are likely to require higher resolutions, at least 1.5^o atmosphere and 2/3^o ocean. NASA data assimilation and forecasting applications demand significantly higher resolution: 1/2^o atmosphere and 1/3^o ocean.

The target atmospheric resolutions appropriate to these applications range from 3 degrees in the horizontal for long-range climate studies, to 1/3 degree for process studies. The vertical resolution for the CSM-2 atmosphere component, CCM-4 will be 30 to 60 levels for the standard model and 100 levels when an active stratospheric model is incorporated. More aggressive resolution targets, such as suggested by the ACPI for support of regional climate studies are not precluded from consideration, but are not currently feasible on the target machines. CCM-4 will carry a set of chemical species and tracers; some focused on physics and transport diagnostics (such as radon and 210-lead) and some to incorporate non-CO_2 gases and their influence on climate prediction (such as sulfate aerosols, methane, nitrous oxide, and ozone).

These numbers emphasize that a performance-portable CSM needs to be able to deal with a wide range of target resolutions, although it is not unreasonable to assume that when running on large numbers of processors the model will be run either at higher resolution or as part of an ensemble.

Our model throughput goals are geared to accelerate the development cycle. For the atmospheric model, we hope to achieve over-night turnaround on a 15-year atmospheric climate simulation evaluating the low frequency behavior and interannual response of the model. Process studies at 0.7 degree resolution will also be targeted. A similar productivity level for the ocean model is a 50-year simulation in 24 hours.

3.2 Target Platforms

DOE production computer platforms, as well as NCAR and other NSF center platforms, define our target architectures and also will provide development cycles for this effort. The DOE centers at National Energy Research Scientific Computing Center (NERSC) and other national laboratories have large high-performance systems computing 64 to 512 nodes, with each node incorporating 1 to 64 processors in a shared memory subsystem. Within a node, shared-memory multithreading (e.g., OpenMP or pthreads) or within-node message passing are both possible. Experience has yet to demonstrate which approach is preferable for a given hardware and model configuration; therefore, it appears prudent to allow for either. Memory systems are multitiered, and cost to access each additional level away from the processor increases. Codes that are able to structure computation and data layout to reuse memory closest to the processor (registers and cache) achieve higher fractions of peak processor performance. Memory reuse is also important to avoid taxing limited intranode memory bandwidth within multiprocessor nodes.

At the present time, CSM must be able to use three specific platforms: IBM SP, SGI Origin, and networked clusters (in particular, the Compaq cluster at NCAR). But it is also important to have the ability to exercise the model on traditional vector supercomputers. The design goal thus will be to maintain performance portability across scalable parallel supercomputers, commodity clusters, shared-memory multiprocessors, and vector supercomputers. Experience with the ECMWF IFS code suggests that this is possible (see Figure 8).

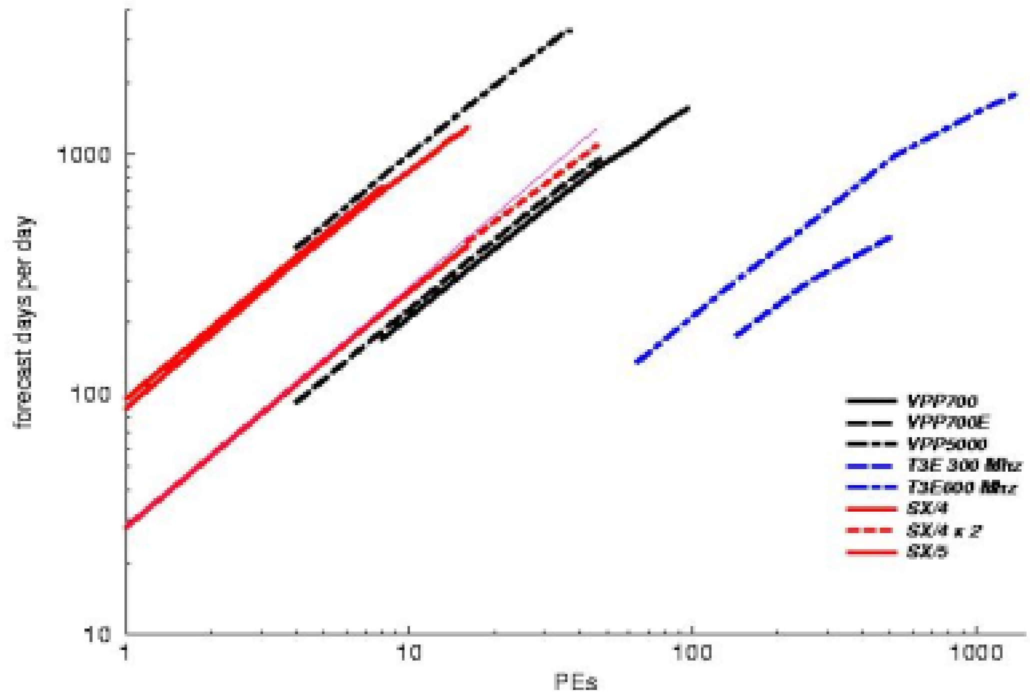


Figure 8: Performance of the Integrated Forecast System (RAPS4) on different computers when run at T213 resolution with 31 levels. The systems studied include several versions of the Fujitsu VPP, NEC SX, and Cray T3E.

Each of the four target architectures introduces distinct challenges for code development.

Scalable parallel supercomputer. The most likely high-performance platforms for CSM-2 in the near term at least are scalable parallel supercomputers such as the IBM SP. These systems feature O(1000) microprocessors connected via a high-speed switch, with these processors grouped in small-processor (4–16) shared-memory clusters. Though distributed-memory message passing using MPI is generally the most portable programming paradigm, it can inhibit code readability and maintainability. The use of a mixed distributed shared memory programming paradigm introduces message passing only at the highest levels in the call tree, and can be hidden from the scientists introducing new parameterizations. Optimization within a shared-memory node using OpenMP for parallelism and vector directives can provide the added parallel performance without affecting code readability. Excellent processor performance within a shared memory node can be obtained if care is taken to manage cache utilization. In the past, the percentage of peak performance achieved on nonvector machines for climate, ocean and weather applications has been less than 10%. With 4 and 8 MB L2 caches becoming standard on scalable supercomputers, it may be possible to realize a higher percentage of peak processor performance, even on the very complex climate models calculations.

Commodity clusters. We can also expect to see CSM-2 used frequently, particularly in university settings, on small to medium-sized "commodity clusters" constructed by connecting high-end microprocessors (e.g., Intel Pentium or DEC Alpha) with fast networks such as Myrinet or Gigabit Ethernet. Performance issues here are similar to those encountered on scalable parallel supercomputers, except that potentially lower network performance and less sophisticated I/O systems may place additional demands on some model components.

Shared-memory multiprocessors. Another common platform for CSM-2 will be small to medium-sized shared memory multiprocessors (e.g., Sun, SGI). A performance-related issue here relates to how coupled models are configured: experience shows that a coupled model structured as a set of independent executables can perform poorly on such platforms, because of high-context switching overheads.

Vector computers. While vector computers are not currently easily accessible to NCAR scientists, the performance and cost-performance of the current generation of vector supercomputers from Fujitsu and NEC remain impressive. Hence, it is important that CSM not abandon support for vector computers. Just as the decision to code exclusively for vector caused problems when emphasis shifted to scalable systems, eliminating support for vector now will prove equally troublesome when barriers to acquiring the latest vector systems are overcome. Moreover, some CSM users currently have access to such systems, and future supercomputer architectures available in the United States may incorporate vector elements. In practice, this means that the model must be constructed to accommodate vector computations along a long, stride-1 data dimension. (Compiler directives and explict vector function calls must also be maintained.) Preliminary results suggest that codes can be engineered flexibly to provide vector-friendly loop structure and storage order along with variable blocking factors for cache optimization (Michalakes et al. 1998; Michalakes 2000; Askworth 1999).

Each platform also implements different input/output (I/O) systems. High-performance parallel I/O has been a weakness of many parallel machine designs. CSM generates large volumes of history and restart data during the course of a multidecade simulation. The I/O subsystems must also be taken into account and a general parallel I/O design, for history output and for restarts, incorporated in CSM.

3.3 Specific Parallel Performance Issues

The development of a performance-portable CSM-2 will require solutions to a number of specific parallel computing issues. While these issues are already being addressed by DOE and other investigators in other research, the timeline under which this project is operating will likely require that we put some effort into the following questions:

- Development of effective methods for achieving high performance on multiprocessors of a SMP node. Specifically, studying the memory locality and interference of cache coherence between different processors, and method to avoid interference between different processor caches. Investigate the performance of OpenMP and other multithreaded parallel execution on the 16 processors on a single IBM/SP SMP node at NERSC.
- Study of message contention issues due to single communication link (the adapter on SP) for multiple processors on the same SMP node. Investigations of mixed MPI and multitasking OpenMP directives
- Investigation of the multithreading programming paradigm, the scoping of parallel constructs, and effective methods (e.g., grouping small messages into one big messages) for improving performance.
- Study of the effects of coarse-grained parallelism on domain decomposition, 1D vs 2D decomposition, etc. Study of parallel transpose/remapping due to shared memory nature of SMP nodes (in Spectral transforms in CCM and remapping for I/O).
- Investigations of basic performance issues on large-scale shared memory architectures, such as process migration, global reduction, optimal operational configuration, etc. These will focus on the specific CSM/PCM models/codes.

4 Proposed R&D Activities

We now describe the work that we propose to perform in this project. As noted above, the primary goal of this work is to develop a performance-enhanced CSM-2 capable of meeting DOE simulation goals on DOE supercomputers. Hence, the bulk of the material in this section is concerned with the techniques that will be used to enhance CSM-2 performance at both the node and multiprocessor levels. Successful development of a performance-enhanced goal will also require that the DOE/NCAR team accomplish a number of subsidiary goals relating to design documentation, performance portability, code readability, and community development infrastructure, and so these topics are also addressed.

Performance portability and code readability are critical because, to be successful, any changes made by DOE researchers to CSM-2 must be immediately folded into the core CSM code base. If this is not done, then the "standard" CSM-2 and the "performance-enhanced" CSM-2 will quickly diverge. But in order for CSM-2 developers to accept DOE modifications, they must be designed to achieve performance portability—that is, good performance across a range of platforms—and must not significantly comprise code readability and maintainability. Performance portability will probably require a mix of performance parametization (e.g., block sizes in physics) and architecture-specific modules (e.g., spectral transform).

Design documentation and community development infrastructure are critical because one side effect of this project will be to expand the community of CSM-2 developers significantly. DOE and NCAR staff agree that successful joint work requires a tighter coupling of NCAR and DOE and NASA developers than has been the case in the past. We believe that this tighter coupling requires both detailed design documentation (so that modules, interfaces, and responsibilities are well defined) and the use of a shared code repository and standard testing mechanisms.

The principal tasks to be undertaken are summarized in the following table.

We expect the PCM-1 and CSM-1 coupled model codes to be replaced with the PCM-2 and CSM-2 codes shortly. These new releases will reflect some merging of the development paths. CSM and PCM will include the same atmosphere, ocean, ice and land surface models. Differences will be primarily in details of implementation. The proposed effort joins the development too late to have a large impact on PCM-2 and CSM-2. We do, however, expect that some of the ongoing development will be useful and easily incorporated into revisions of these codes. As each stage of the CCM-4 is completed, the older version of CCM in PCM-2 and CSM-2 can be swapped out. Similarly, specific optimizations of the flux coupler can be inserted in both models. The expectation is that the PCM and CSM models will be fully merged at version three as a result of this proposal.

4.1 High-Performance Atmosphere Model

The PCM and CSM coupled climate models share a common atmospheric component, CCM. Improvements to the atmospheric component can thus be expected to have an immediate impact on both coupled models. As CSM and PCM merge, we are mindful that ongoing production runs performed with PCM get a positive benefit from the parallelism improvements proposed here and that the scientific development of CSM continue while its simulation capabilities are accelerated.

Unfortunately, development of CCM-4 must initially be slowed by this collaboration even with extra resources added to NCAR. Knowledgeable NCAR staff must be diverted to interact with the DOE investigators. New staff simply do not have the expertise required, and general experience at NCAR and other similar atmospheric modeling centers such as UKMO/Hadley Centre is that it takes 6 to 18 months to fully spin up new people; the short range of 6 months is extremely rare. However, we expect to more than regain this initial setback during the period of this proposal.

The strategy proposed for the development of the atmospheric component will be to begin with a software engineering design for CCM-4. This design will be extended to meet near-term goals and an implementation plan put in place to incorporate scalable parallelism and modest encapsulation goals in a single source CCM-4. The design will then be revisited and extended in a more complete “open design” process once CCM-4 is complete. This strategy offers the shortest path to single source, with the lowest adverse impact on CCM-4 development and the highest productivity impact on both PCM and CSM applications.

The preliminary design and implementation plan will seek to increase the modularity of the atmospheric model by clearly encapsulating the dynamical cores (“dycores”). This is a continuation of the work already begun at NCAR. Incorporating this strategy in the initial design is important for three reasons. First, by clearly defining the interface in the atmospheric model, code optimizations and parallel decomposition strategies for the dynamics can be developed independently and optimally of other model components (e.g., physical parameterizations). Second, parallel constructs will be more isolated so that the model can be readily ported and adapted to new platforms. Third, research into new dynamical cores can be accelerated, since multiple groups can contribute and since the differences among the climates produced by different dynamical cores can be diagnosed in the common framework. The realization of this design goal in an early version of the atmospheric model will be a great step forward for those interested in software engineering and parallel performance. If it is realized in an efficient manner, it will allow the CSM code framework to carry the time-tested methods along with select novel methods, until “winners” emerge. It is also entirely plausible that different applications teams may favor different dynamical cores.

The need for “swappable dynamics” has been recognized and recommended by a grass roots movement to promote a national modeling infrastructure. The design work proposed here shares common goals with the Common Modeling Infrastructure Working Group (http://nsipp.gsfc.nasa.gov/infra), but is more focused and will contribute as an example of the kind of design that can be done without sacrificing quality of simulation or performance.

A scalable parallel design using a mix of distributed- and shared-memory parallelism is important if production cycles are to be utilized on the target platforms. The design must allow for adequate granularity in the parallel decomposition to effectively utilize the scalable, high performance computers. Of course, the design must pay special attention to effectiveness on machines emphasizing low-resolution ensemble runs for DOE applications. Decomposing the spatial domains in two or more directions can yield the required granularity. The current assumption of this programming model is that the "inner" directions will be handled in shared memory using the standard OpenMP multitasking directives and the "outer" directions of the decomposition will be implemented for distributed-memory message passing. Thus a two-dimensional decomposition for the atmospheric code could be achieved by decomposing the latitudes in distributed memory and within each latitude slice using shared memory. Other decompositions may yield better performance, especially where load balancing is of concern. The parallel decomposition can be different for different dynamical cores. A different decomposition for the calculation of the physical parameterizations (column physics) may also be advantageous. Certainly, to achieve performance portability across the target architectures, a degree of flexibility is a necessary design goal for our project.

Finally, the design must include model validation and code testing procedures. By model validation, we mean validation between different platforms where it is expected not to produce exactly the same numerical results. One example of such a validation procedures is to look at the rate of separation between results started from the same initial conditions (see Rosinski and Williamson 1997). Code testing procedures must include checks to see that the code is working as expected. These may include performance expectations, numerical results, reproducibility and reconfigurability. The inclusion of such procedures in a design document is important so that everyone understands what the rules are and how much leeway programmers have in making code modifications.

The design document for the atmospheric component of CSM-2 should be thought of as a dynamic work in progress. We will start with a preliminary and not very extensive document, which, over the period of this proposal, will evolve into a more comprehensive and precise design specification. A process will be set in place, in conjunction with the CSM Software Engineering Working Group and NCAR management, that will allow for design reviews both internally and externally. With DOE, NSF and NASA participation in this project, an "open" design process will result with the design document as the public deliverable. As the design evolves, we expect to identify further encapsulation in the software design, which will require interfaces to be documented and implemented. As these are more clearly defined, a class library, or module structure, will emerge; this is the pattern in many large software projects.

We expect this evolution to take place, but we are not currently in a position to prescribe it. Instead, what we propose is to get involved immediately with the development of the CCM-4 model and code. The specific steps and design we will follow are outlined in the following implementation plan.

4.1.1 Accelerated Development of the High-Performance Atmospheric Model

Since this software engineering project starts in the middle of the CCM-4 atmospheric climate model development at NCAR, we have planned a development path that will interface with the ongoing development with minimal disruption. Code restructuring will be performed incrementally, with three phased developments. The first phase is a design study, remapping of data structures and restructuring of the code for the calculation of physical parameterizations, the column physics. Second, the dynamical core interface will be formalized and implemented in CCM-4. This work will make it possible for parallel implementations of different dycores to proceed independently. The third phase will be the parallel development of three high performance dynamical cores and integration with the atmospheric model. After the first phase is complete, we project that the second and third phases will require minimal diversion of the NCAR development group from the refinement of the content of the model. What will result is a single-source CCM-4 that meets the preliminary design goals for a high-performance atmospheric model.

The interface of the dynamical cores with the CCM-4 is described in Figure 9. This interface description shows the two major components of the atmospheric code: the physics parameterization package, also called the column physics, and the dynamical core. In CCM-4, all parameterization packages will provide tendencies and be given state information. The dynamical core will also provide tendencies being given state information. The interface handles the data structures required for good performance within the dynamical core and the physics package, by transposing the data. If the dynamical core operates on the same data structure as the physics, the transposes are unnecessary. The data transformations are necessary if the dynamical core operates with a different grid structure from the physics. Thus, the transformations map between method grids, and the transposes map between parallel decompositions of the data. The design is very clear in explicitly isolating these two functions during the basic timestep. The time split interface and the process split interface are two numerical methods for advancing time; both will be implemented.

Figure 9. Dynamical Core Interface for the Basic Timestep

1. **Prognostic variables stored in form convenient for the dynamical core**
2. **Copy of data *transformed* to unstaggered grid for parameterizations**
3. **Data *transposed* to provide columns for parameterizations**
4. **Parameterization package provides tendencies (specified variables passed to history buffer)**
5. **Tendencies *transformed* back to dynamical grid**
6. **Tendencies *transposed* to decomposition of dynamical core**
7. **Tendencies added**
   - **to prognostic variables for the time-split interface,**
   - **passed to dynamical core for process split.**
8. **Dynamical core provides tendencies (this may involve additional data transpositions; specified variables passed to history buffer)**
9. **Prognostic variables updated (prognostic variables passed to history buffer; restart variables passed to restart buffer)**

Three dycores will be implemented in CCM-4, the Eulerian-Spectral, the Semi-Lagrangian-Spectral, and the Lin-Rood Lagrangian-Finite Volume dynamical core. The Lin-Rood method has been under development at NASA and will be extended in the context of this proposal for high performance parallel architectures. Initially, we propose work on all three of these dynamical cores; we will need to carefully reassess the level of effort devoted to each as the CCM user community gains experience in their use.

Next, we describe the proposed work for the second phase of development on each dynamical core.

4.1.2 Eulerian–Spectral Dynamical Core

The Eulerian-spectral dynamical core will be restructured to eliminate the strict dependence on a latitudinal parallel decomposition. The parallel algorithm for the spectral transformations will exploit both a distributed-memory (MPI) parallelism as well as a shared-memory (OpenMP) parallelism.

First, the data transposition software will be customized and optimized to map distributed dynamics data to distributed physics data structures (see below for segmented physics data layout). The transpose software will be adapted from the methods used in the 2D parallel decomposition of the PCCM-3 developed by Drake et. al. (1995), to a generalized decomposition. These data transpositions will take account of the physical space data as well as Fourier and spectral coefficient data.

The dynamics data decomposition will focus on a segmented implementation (same as physics) and a level-fields decomposition that maintains horizontal field coherency for the transforms. The segmented implementation will transpose and decompose data to calculate the Fourier transforms, but do a distributed sum on segments for the Legendre transform. That way the FFTs can effectively use library software on a computational node and take advantage of the peak shared-memory performance of the platform. The segment size will be adjustable to hit maximum cache performance as well as to expose parallelism on a reduced grid for multitasking and message passing. The standard 1D parallel decomposition of CCM-3 will thus be a special case and vector lengths that support good vectorization will be a runtime option.

The decomposition by layers will focus on shared-memory parallelism of FFTs and Legendre transforms within a horizontal level on a reduced grid. This is a natural and easily exploitable decomposition for spectral algorithms on clusters of shared-memory machines. It was not an option in the design of PCCM-2 and PCCM-3 because the number of levels (18) and the lack of effective shared-memory nodes forced the use of message passing only. But the minimum design point for the CCM-4 is 30 levels and several more fields. Shared-memory nodes with 4–16 processors are the order of the day, so we expect this decomposition to be efficient for the parallel spectral transform.

Both the segmented and the layer data decompositions work well with a “reduced grid” for the spectral dynamics. In fact, the generalization from a simple 1-D or 2D parallel decomposition is necessary for proper load balancing of the computation on a reduced grid. In both decompositions, the number of gridpoints calculated by a compute node can be evenly distributed. The segmented decomposition can use segment size to accommodate fewer grid points per latitude as the poles are approached. With a layered decomposition the number of points is exactly balanced.

After the transpose and new parallel spectral algorithms are in place in the dynamical core, the semi-Lagrangian transport of the tracer fields will be addressed. This involves the same considerations and strategies as for the semi-Lagrangian-Spectral dynamical core. Exploiting this overlap, we hope to implement two dynamical cores for the effort of something less than two.

4.1.3 Semi-Lagrangian-Spectral Dynamical Core

The semi-Lagrangian parallel algorithms are shared between all three dynamical cores. We will coordinate the development and implementation of supporting routines and libraries. For horizontally decomposed data, like the segmented physics structure, a set of halo update routines will be adapted from the PCCM-3 and the Lin-Rood developments. The halo region contains data from “neighboring” nodes that is required to complete the particle-tracking step of the semi-Lagrangian algorithm and for interpolation of advected fields. On a reduced grid, the halo region update requires a level of indirect addressing since the neighboring points are no longer identified simply by their indices. But the assumption of tensor product grids and interpolations based on these grids makes this tractable. The halo region must be updated every timestep, but by dynamically monitoring flow directions and magnitudes, the amount of communication required can be minimized.

The decomposition by level (and fields) described in the Eulerian-Spectral dynamical core, will also work well for the semi-Lagrangian aspects of the dynamics. Halo regions will no longer be required for the horizontal directions and communication costs of vertical advection are minimized (or non-existent for the floating Lagrangian control-volumes of Lin-Rood). The polar region likewise requires no special treatment, and shared-memory parallelism can be easily implemented for the calculation within a layer or (local) collection of layers.

4.1.4 Lin-Rood Dynamical Core

The CCM-4 model now includes the recently developed Lin-Rood dynamical core (Lin and Rood, 1996; Lin and Rood, 1997; Lin, 1997; Lin and Rood, 1998; and Lin and Rood, 1999). The basic algorithms are derived and evolved from the modern high-resolution finite volume algorithms pioneered by van Leer (1977) and Colella and Woodward (1984), which are one-dimensional algorithms designed primarily for astrophysical and aerospace engineering applications requiring the resolution of sharp gradients (e.g., shocks). Details of the Lin-Rood dynamical core can be found in a recently produced algorithm theoretical basis document (DAO 2000). In summary, its unique attributes are the following:

- terrain-following “floating” Lagrangian control-volume vertical coordinate with monotonicity-preserving mass-, momentum-, and total energy-conserving mapping algorithm to the fixed Eulerian reference coordinate;
- genuinely two-dimensional, physically based, conservative semi-Lagrangian transport between two bounding Lagrangian surfaces that define the finite control volume; and
  - accurate finite-volume representation of the mean terrain with accurate and physically consistent integration of pressure gradient force for the terrain-following Lagrangian control-volume.

The semi-Lagrangian aspects are especially important in the east-west direction by enabling accurate simulations near the polar areas without the need for large reductions in the time step.

The development of the Lin-Rood dynamical core will provide a scalable parallelization based on a 2D (latitude-vertical) domain decomposition of the physical space. Given the flux form semi-Lagrangian formulation in the east-west direction and the floating Lagrangian vertical coordinate, blocks of latitude and altitude regions will be distributed to computational nodes. The entire length of cells on a longitude will be contained in the shared memory of a computational node. Shared-memory (OpenMP) parallelism will be exploited along this longitudinal direction.

Transposing the data for this decomposition has been shown to be efficient and software to achieve these tasks can be jointly developed with the other two dynamical cores. The software for updating of the north-south direction halo regions can also be developed jointly with the other dynamical cores.

4.1.5 Parallel Physics: Data Decomposition and Code Rework

The parallel decomposition of the parameterization package, the column physics, is a key to high performance of the atmospheric model. Since each vertical column of the atmosphere can be computed independently, there is abundant parallelism for a scalable algorithm. As chemistry and more comprehensive physical parameterizations are included in the model, this parallelism only becomes easier to exploit.

Two considerations are very important, however, and will be addressed as part of this proposal. First is the issue of node performance on the physics computation. At an appropriate time in the development of CCM-4, we will rework the parameterization package to support a general segmented data decomposition. The required rewrite will change the parameterization package so that it works on a collection of columns, with inner loop indexing running from an ibegin to iend. Currently, the inner loops assume that an entire latitude slice is being processed with the ending index variable to accommodate a reduced grid. We will add the beginning index construction. The current choice is excellent for node performance on vector machines but requires an extremely large cache for the types of computer architectures prevalent within the DOE community. The segmented data decomposition is thus a generalization of the 1D and 2D parallel decompositions, containing both as a special case. In addition, it is a simple decomposition in that all the details of the column location can be hidden. It will thus simplify the coding style and readability of the code. Cache utilization and local shared-memory multitasking will be supported with this decomposition of the physical space so that the scientists developing parameterizations will not need to multitask their parts of the code. These basic code modifications will be done as soon as reasonably possible given the evolving parameterization code. Since this portion of the code consumes a large fraction of the total compute resources it will be best to address it early in the planning and development process.

The second consideration is load balancing. Especially for reduced grids a general data decomposition is required. Since the local node will call the parameterization package with an arbitrary collection of columns, the column assignment can be made to effectively balance the load among the computational nodes. The short wave radiation calculation is an example of the type of load imbalance that occurs. By assigning an equal number of day columns to the nodes, the computational load can be equalized. This assignment will be accomplished in the context of the data transpose steps described in Figure 1. The inclusion of load-balancing transpose routines will be accomplished in the third step of our phased development.

The design of the column physics could require that location (lon,lat) is passed to the physics rather than assumed based on index values. This may be necessary to support the general load balancing and segmentations we are proposing. These decisions will be made and documented in a preliminary design that will guide this remapping of the physical parameterization code.

4.1.6 Atmosphere - Land Surface Interface

In CSM-1, the land surface model used the same grid as the atmosphere model and was tightly coupled with the atmosphere model. Recently, NCAR staff have reworked the internal parallelism of the land model and its interfaces to the atmosphere model and coupler. The interfaces are now nearly identical and the code is unified. The decomposition of the land model is entirely independent of the atmosphere, even when both are hybrid MPI/OpenMP parallel with internal coupling. The price is that data is all gathered prior to transmission in each direction, as has always been the case through the coupler, but not for the internal coupling. This decision was required both for maintainability and for scaling of the land model. How best to deal with these interface and coupling issues still needs to be carefully rethought, but the starting point is different from before. In future work, we need to eliminate the requirement for a gather before communication with the coupler, for all components and regardless of the coupling method.

4.1.7 Coordination Plan for the Atmospheric Model

For the atmospheric model development and integration with CCM-4 we have agreed to the following responsibilities among DOE and NSF developers.

NCAR	Dycore interface, version control	Williamson
ORNL	Eulerian-Spectral and Semi-Lagrangian-Spectral dynamical core, transpose algorithms	Drake

LLNL	Lin-Rood dynamical core, halo algorithms	Rotman

The Lin-Rood dynamical core development will be coordinated with existing NASA and NCAR projects. Project developers will coordinate code development using a source code version repository managed by the CCM-4 core group at NCAR. Additional support for the NCAR tasks is required, particularly to resolve or prevent the code conflicts that will result from the addition of two more development threads. A tight coordination strategy can minimize the impact on the science development of CCM-4. The atmospheric model development project outlined in this proposal will test and explore how best to achieve this required coordination for developers working at a distance.

The testing of code before checking into the repository must be clearly defined at the beginning of this project as part of the preliminary design. It is intolerable to disrupt ongoing development activities to "clean up" another developer's mistakes. This is the flash point for a joint development project and must be addressed from the beginning. We will schedule regular teleconferences and "check in parties" among the developers to ensure that goals are being met.

In addition, we plan quarterly meetings of the atmospheric model interagency collaboration.

4.2 High-Performance Coupler

We have adopted the CSM coupler as an early priority for DOE/NCAR collaborative effort because of the critical role it plays in CSM execution and because it represents, in its current instantiation, a significant performance bottleneck. Our goals in this area are twofold:

1. To work with NCAR to design and develop a next-generation coupler that supports efficient sequential and distributed execution, so that a user can select the model/coupler configuration that makes the most sense for a particular problem/platform configuration.
2. To address, specifically, performance problems associated with various coupling functions: in particular, regridding operations and load balancing problems.

In the rest of this section, we present several flux coupler design concepts that we believe can, in combination, allow us to address these goals. These concepts derive from the observed combined limitations of CSM and PCM. We have identified these elements of a successful design that need significant innovation: solving the handshake problem through an object oriented design, developing scalable parallel regridding software, developing a general coupled climate model design that allows overlapped MIMD execution of SPMD components, with execution on flexible numbers of processors. We assert that these goals can be achieved with either a single or multiple executable design.

We believe that the tasks listed below can productively be approached in a two-step process. In a first step, the current coupler will be optimized to improve scalability of the current CSM. Concurrent with optimizing the current CSM coupler, a DOE/NCAR team will work to develop a design for a next-generation coupler that is intended to have the following properties:

- Support for both sequential and distributed execution, and probably also various hybrid configurations in which some components are distributed and others are not.
- High-performance scalable parallel implementation of performance-sensitive regridding and communication operations, to enable CSM execution on large numbers of processors.
- Flexible configuration enabling run-time rather than compile-time specification of processor allocations (in the distributed case).
- Support for standalone execution of individual component models, with the coupler being used to manage boundary condition input from files.

4.2.1 The Handshake Problem

To solve the handshake combinatorics problem described above, the coupler must be described by an effective object model that separates out the concepts of components, fields, and grids. With the proper abstractions it becomes clear how different flux coupler methods are related and how they interact. For example, time-averaging and flux calculations become methods that operate on different fields that reside on the same grids, whereas regriddings transform a field from one grid to another. An effort is already underway to provide the rudiments of a grid describing framework for the CSM history file subsystem. We propose to collaborate with this effort to develop the right data abstractions for the flux coupler.

Additionally, to avoid unreadable complexity of the intermingled interactions with multiple components, the coupler design should also segregate its interaction with each model component into separate modular pieces of code. This would allow easy experimentation with different modes of coupler interaction.

4.2.2 Parallel Regridding Software

Parallel regridding software is the machinery that must stand behind this new object-based flux coupler design if it is to succeed. In the current parallel regridding algorithm in PCM, each node migrates the source points needed to perform the regridding to the destination patch on each node. The regridding is then performed in parallel on each node. Timings of the current PCM flux coupler on the IBM SP-2 indicate that the regriddings effectively stop scaling beyond 16 processors. As a result, the percentage of time that PCM spends in the flux coupler grows from 8% to 18% as the number of processors scale from 8 to 64 processors, respectively. The cause of the scaling problem is well known: it resides in the load imbalances caused by variations in the number of migrating points required by each node. If we consider the regriddings to be sparse matrices that map one grid “vector”' into another, we can see this imbalance by binning the number of points in each row of the sparse matrix mapping (see Figure 10).

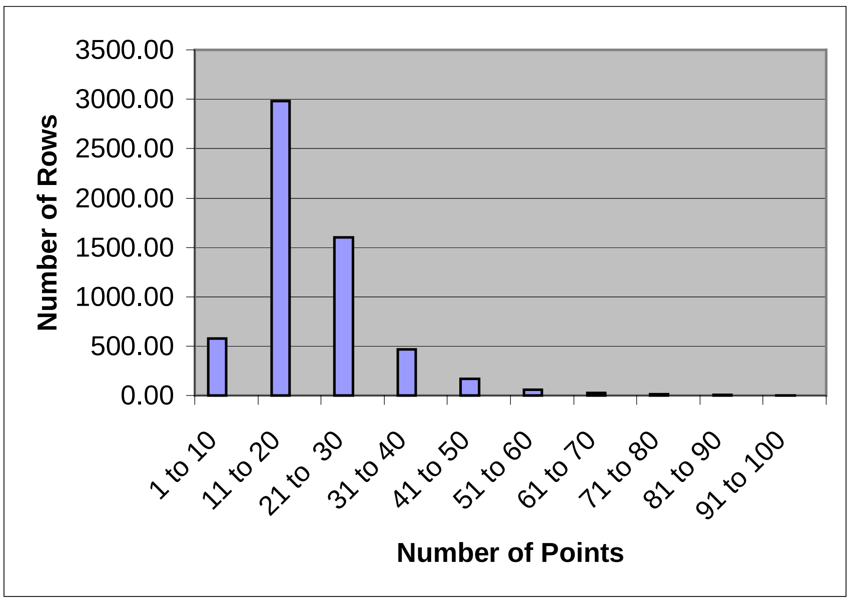


Figure 10: Histogram showing the sparsity structure for a conservative mapping from a 2/3° ocean to a T42 atmosphere.

These data show that the amount of regridding work varies, for some points on the sphere, by as much as an order of magnitude. We can use the following techniques to address this load imbalance. First, the regriddings should take advantage of a hybrid threading under message-passing approach, so as to allow dynamic load balancing of regridding work between threads within SMP nodes. Second, third-party load-balancing software products should be evaluated as potential solutions to the static load-balancing problem. Ideally, one of these existing software packages would support, or could be adapted to automatically generate, the irregular decompositions that would minimize communications and optimize the scalability of the regriddings.

4.2.3 Implications for Model Structure

It is impossible to develop a design of the flux coupler without understanding the overall parallel execution model of the climate system. For this reason we present here some the design considerations for the entire climate system.

Concurrent Execution of Components. The potential advantage of exploiting component level MIMD parallelism of the multi-component climate system represents a level of parallelism too important to omit from it a general design. As noted earlier, asynchronous execution introduces extra communication by separating the flux coupler from the components. However, if these communications can be converted into the parallel data motion to and from irregular domain decompositions that effectively load balance the regriddings, this data motion is no longer superfluous.

Component execution on flexible numbers of processors. Model components differ in the resolution, implementation, and computational cost, and so may differ dramatically terms of scalability. Models often cease scaling, slow down, or fail to run at all if pushed beyond certain limits of parallelism. Therefore, models cannot be required to execute on the same numbers of nodes.

Automatic load balancing. Load balancing represents a significant obstacle to the effective and efficient use of CSM when using a distributed coupling strategy. While run-time specification of processor allocations makes it easier for the user to tune allocations to avoid load imbalances, in practice we will find that the sensitivity of load distribution to such issues as choice of components, parameter settings, resolutions, processor decompositions, and system architecture will make the selection of appropriate processor allocations a daunting task, certainly well beyond the interest or knowledge of an average user. A solution to this problem is to incorporate automatic load-balancing techniques into CSM.

Single executable vs. multiple executable. A sequential time-split or asynchronous time-lagged model integration scheme can be implemented in either a single-executable or multiple-executable design. Therefore, the time integration issue, as well as most other design issues, can be safely considered separately from the executable issue. There may be circumstances in which a single-executable design addresses some computer system software limitation. Otherwise, the single-executable design seems preferable since modeling codes remains separate and self-contained entities and therefore less prone to unintended interactions. Unfortunately, experience indicates that this issue often has more to do with modeling group politics than software design.

4.2.4 Planned Activities for Flux Coupler Optimization and Development

CSM, PCM, and NCAR Scientific Computing Division (SCD) groups are currently engaged in discussions concerning coupler design issues relating to (1) structuring the flux coupler software to work in a distributed shared memory environment and (2) determining and synchronizing the coupling strategy used for the component models. NCAR and CSM developers are actively debating both of these issues. The work proposed here seeks to contribute primarily in the first set of issues, by developing components that will be utilized in any successful design. Hence, we propose to implement the functionality of the coupler as a set of modules that

- Average data over time intervals and keep track of “model time” for synchronization
- Compute fluxes based on state information provided by the component models
- Manage communication between distributed parallel components
- Perform regridding and interpolation functions

This functionality can be brought forward from the PCM-2 and CSM-2 flux coupler designs, modularized and placed in an object-oriented framework according to the design document generated in conjunction with the proposal. The components can be flexibly arranged according to the interface design and synchronization strategies of component models. Since the interface design is the subject of ongoing discussion among component model developers, we will coordinate closely with that effort. The flux coupler software infrastructure can progress somewhat independently of the interface design questions.

There will be two phases in our work on the flux coupler. The first phase will be detailed performance analyses of both the PCM and CSM codes, with particular attention paid to their respective flux couplers. This phase will begin immediately, and will extend through the first several months of this project. The second phase of the project will be the design and implementation of the next generation coupler for the CSM. There will be considerable overlap in the timelines for the two phases.

The rationale for this approach is twofold. First, the CSM group is in the early design stage of the next generation coupler. Their current estimate is that the new coupler will not be completed and ready for production use until the end of 2000. Addition of DOE personnel to this product will most likely not significantly accelerate this process (Brooks, 1995). Involvement of DOE personnel with their considerable expertise in software engineering and parallel computing will, however, result in a better product. Secondly, the current need for performance enhancements to both the CSM and PCM systems requires prompt relief, which may be achieved through performance analyses of both the CSM and PCM systems. Such performance studies should yield information regarding the effects of coupler architecture on performance, and also identify optimizations that may be implemented in the near term.

Phase I: Performance Studies and Optimization: Our first step will be a complete inventory of currently available performance data for both the CSM and PCM systems. (See http://www.cgd.ucar.edu/ccr/bettge/PCM/CUG1999.html for PCM coupler performance results). These data will be analyzed to determine what additional performance studies are needed to: 1) elucidate the performance characteristics of each coupler architecture; 2) identify potential bottlenecks in each of the couplers. The understanding of the performance of each coupling strategy on the platforms we intend to support will have a profound impact on the design for the new coupler. The identification of any performance bottlenecks will allow us to formulate near-term plans to improve the single processor performance and scaling of the current PCM and CSM couplers.

The performance data we shall seek are: 1) detailed profiles for the PCM and CSM systems for each of their coupling strategies; 2) load balance characteristics for each of the systems; 3) computation versus communication ratios; 4) cache usage statistics.

Phase II: Design and Implementation of the Next Generation Coupler: There are two main design issues for the new coupler: (1) top –level coupling strategy and (2) identification of low-level utilities or "building blocks" used to implement the coupler. The top-level architecture of the next generation coupler will be designed in response to CSM scientific requirements and the result of our Phase I performance studies. The low-level building blocks include: parallel data structures representing the fields and fluxes exchanged between the coupler and component models; routines to create, destroy, and manipulate these parallel data structures; efficient regridding algorithms (i.e., sparse matrix-vector multiplication implementation of the grid interpolation schemes); communications infrastructure, including data structures to describe domain decompositions, and communications routines that transparently use these structures to simplify the exchange of data between the coupler and component models; load balancing mechanisms. The design of the building blocks will be the first stage of the design process for the new coupler. These building blocks could be prototyped and tested by attempting to use them in the current flux couplers. Where possible, these building blocks will include optimization strategies discovered during Phase I.

As the top-level design of the coupler(s) solidifies, we will begin implementing it using the low-level building blocks.

4.3 Improvements to Coarse-Resolution POP Simulations

We noted above that the performance of POP's barotropic solver can hinder scalability at coarse resolutions. Initial tests have shown that an altered decomposition can produce improved scaling. The alteration is to continue the baroclinic solution in a 2D domain decomposition, but make use of a reduced set of processors for the barotropic solve. Since the baroclinic solve involves a 3D solution, while the barotropic involves only a 2D solution, the solution is better balanced. These initial tests were designed as proof of concept. Under this proposal we will investigate this concept further by implementing libraries in Argonne's Portable Extensible Toolkit for Scientific Computation (PETSC) framework to enable efficient decompositions.

In addition, work is under way to investigate an implicit solution technique whereby the locality of informational exchange inherent in many physically based PDEs is used to minimize global communications. This algorithm will enable long time steps and scalable solutions. We will investigate this technique and, if appropriate, apply it to the POP ocean model.

4.4 Parallel I/O

Currently, the I/O system on CSM has two tracks, the history files and the restart files. The history files are written in netCDF format, while restart files are written using Fortran unformatted modes. In the current SMP implementation, each model component handles its own history files and restart files independently (with the coordination of the coupler).

Parallel I/O techniques are required within CSM-2 for performance reasons. Important design goals for parallel I/O include (a) portability, since I/O systems on different computer platforms differ substantially;

(b) high efficiency, due to the high output of data during century-long simulations; and (c) convenience, enabling minimum changes from the existing CSM I/O framework, adaptability to changing environment, elimination of post processing, and so forth.

In investigating parallel I/O, we can build upon experience gained developing parallel I/O techniques for the Modular Ocean Model (Ding and He 1999). The main idea developed there to achieve both portability and high I/O efficiency is to (1) remap distributed 1D, 2D and 3D fields (arrays) to/from a subset of designated processors; and (2) read/write the remapped array as contiguous blocks of a 1D array in collective I/O from/to a single file.

In this single file approach, data files are written as if they are processed in sequential environment, irrespective of how many processors will access them. This approach allows the output history and restart files to be directly analyzed and visualized in any other workstation environments, without extra file conversions. It is adaptable to a changing computing environment. For example, a simulation may proceed for some time on 256 processors. After a checkpoint, the simulation can restart on 128 processors, since the restart files were written in natural order, irrespective of the number of processors that wrote it.. The approach also is portable to any other platform, since a standard UNIX file interface is used, with the only requirement being that multiple processors can write contiguous blocks into disjoined segments of the same file in parallel.

An efficient remapping algorithm and implementation has also been developed. Since the communication bandwidth is typically about ten times faster than I/O bandwidth, remapping does not degrade total efficiency.

This parallel I/O strategy has been successfully implemented for part of the MOM3 ocean model. Figure X shows the total I/O time including file open/close/writing/remap times for four 3D fields and five 2D fields for two different resolutions, using netCDF format. It is efficient and scales well. Assume we run at 10-simulated year per wallclock-day turnaround time for the T42 resolution for atmosphere and 2 degrees for ocean. This implies that history files of monthly averages must be written every 12 minutes. A simple estimate of writing the required roughly 300 MB data into history files would take about less than 10 seconds. This amounts to about 1% of the total time on I/O. The restart file can be written much less frequently and thus is not an important factor in determining the simulation rate. When the resolution doubles, the I/O requirements increase by a factor of 4, which is still manageable.

The large number of files simultaneously opened by different component models and files of different natures substantially increase the complexity of the I/O system. There is an effort at NCAR to develop standards including data objects and classes. Our parallel I/O development is a complementary effort, since our work will be done at a lower level closer to hardware. Specific tasks to be undertaken here include the following:

- Evaluation of I/O performance of the current CSM on scalable parallel computers. Current I/O for each components in CSM for PVP system need to be modified for distributed memory system.
- Integration of parallel I/O techniques into the CSM framework. Jointly with NCAR, identify/develop the most convenient and portable unified I/O interface/approach for all component models.
- Joint work with NCAR staff to compare and contrast results gained in this work with an alternative NetCDF-on-HDF5 approach, with the goal of identifying the best approach to parallel I/O for CSM-2

5 Code Development Methodologies and Infrastructure

The success of this project will depend critically on the coordination of the many scientists and software engineers from DOE labs, NCAR, and NASA. This coordination will require a clearly defined and effective software development process, so that 1) the improvements described in this proposal are incorporated into the CSM code, and 2) the CSM group assumes maintenance and future responsibilities for the code. The elucidation of a clear and practical development process that can be merged with the CSM group's current development practice is an important deliverable of this project.

The five basic elements of our development strategy are as follows:

1. The development of a CSM Software Developer's Guide, which describes a clear set of coding practices and standards.
2. The development of a series of design documents, which articulate both overall code structure and the design of specific components.
3. The definition of a jointly agreed upon staged software development cycle that includes technical reviews and quality assurance (QA) procedures.
4. The use of a common code repository.
5. Communication and archival mechanisms that will keep the many developers involved in the project aware of what has been done and what needs to be done

5.1 CSM Software Developer's Guide

We propose to work with NCAR staff to create a "CSM Software Developer's Guide." This document will contain a description of the accepted sequence of review and testing stages in the development of a software component, as well as conventions for naming, code formatting, error handling, argument ordering, configuration management procedures, and so forth. Our goal in producing this guide is to improve the consistency and quality of code produced by a diverse team of developers.

The Developer's Guide will reference or describe software language standards. An example of such standards for Fortran 90 are those established by the European numerical weather prediction community (available online at http://www.meto.gov.uk/sec5/NWP/NWP_F90Standards.html).

5.2 Design Documents

A logical first step in software development is preparing a statement of what the software must do. This requirements list can then be analyzed and used as the basis for a software design. A software design document contains both a summary of the requirements and the design specification.

Overall design documents will communicate the desired high-level structure for CSM and CCM. These documents will include a description of major data structures and calling tree.

Specific design documents will be prepared for individual code components. These documents will describe the nature of the code and will state in detail the code interfaces and data structures.

Design documents are a natural basis for documentation and will be prepared in a fashion that makes them easily convertible to this purpose.

5.3 Staged Software Engineering Development Cycle

We propose to introduce a staged development cycle that will create regular checkpoints to ensure that development activities are coordinated and that code meets quality standards. Technical reviews and several levels of code testing are key aspects of this cycle.

Each software component (coupler, I/O library, dynamical cores) will step through the following stages.

1. *Write and review a requirements list.*

This proposal may be a sufficient statement of requirements for the overall design document. Preparation of separate and more detailed requirements lists for individual components will help to coordinate developer activities.

2. *Write and review a design document.*

We anticipate that the development of a design document will be accompanied by the creation of code prototypes. A summary of requirements will be folded into and updated with the design document.

3. *Write and review code.*

Code will be written in accordance with standards laid out in the Developer's Guide. The code review will consist of a small number of developers who carefully inspect a piece of code. It is an excellent way to identify mistakes and inefficiencies at an early stage of development.

At this stage the code should pass the first and most basic level of testing, which is whether the code can be compiled and built on the platforms that we intend to support. This testing can be automated, using the same approach as other development groups (e.g., PETSc, Globus).

4. *Unit test code (test stand-alone).*

Thorough testing of code before checking into the repository must clearly be part of the development process. It is intolerable to disrupt ongoing development activities to "clean up" another developer's mistakes.

5. *Integrate and validate code.*

Integration means incorporating a code segment into a working version of the CSM and ensuring that it correctly interoperates with other portions of the code.

Many of the changes we will make to the CSM code will involve optimization and reorganization of the source code, and thus should produce changes that are of round off magnitude. Given an appropriate definition of "round off," testing of these changes could be automated. Profound algorithmic changes, such as a new solver for the primitive equations or new physics parameterizations, require additional testing to ensure the model will run well over long-term integrations. This level of testing is called validation.

6. *Update the design document and convert it to documentation.*

5.4 Formal Technical Reviews

The fact that the CSM is a community model—and that the next generation of the model may be used in an operational forecast setting—provides motivation for careful software quality assurance. The natural approach is peer review through the formal technical reviews (FTR) or walkthroughs (McConnell 1993) mentioned in the previous section. A review or walkthrough is a moderated meeting in which requirements, software design, or a piece of code is examined and discussed.

The three types of reviews have different participants:

- scientists review requirements
- scientists and software developers review the design document
- developers review code

Possible outcomes of a review are:

- schedule a date for release of a final version incorporating reviewer comments
- schedule a date for another review

It will be helpful to have a coordinator participate in all reviews to flag redundancies and conflicts.

5.5 Developers' Repository

Since scientists and software engineers from both NCAR and DOE labs have experience with Concurrent Versions System (CVS), we intend to use it as the basis for a common repository and version control. NCAR. From a management point of view, CVS is desirable because there are both Windows and Macintosh implementations that include a graphical user interface (see http://www.vincvs.org). CVS is also free, which makes it appealing for university researchers. It is downloadable by anonymous ftp at http://www.sourcegear.com/CVS.

5.6 Communication mechanisms

We propose numerous mechanisms for keeping developers apprised of the current state of the project and its future directions. A development website will archive all reports, meeting minutes, test data, and other technical information relevant to the project. Mailing lists will be used on many aspects of the project and email traffic from these lists will be archived at the development site. As the project matures, we may consider implementing a bug-and-change tracking system such as GNATS (see http://www.gnu.org/software/gnats/gnats.html), where bug reports can be filed and will be routed automatically to the developers responsible for the components in which bugs have been detected. In addition, we will work in close coordination with NCAR staff to identify a range of wide-reaching communication mechanisms to improve understanding and knowledge of the role that software engineering plays in climate model development These mechanisms may include an NCAR seminar series, a software engineering website, and a forum that holds regular meetings, much like the "PDEs on the Sphere" workshop series, which has played a significant role in communicating advances in numerical methods.

6 Management and Coordination

The tasks to be undertaken in this project are clearly defined in Sections 4 and 7. Day-to-day work on these tasks within the DOE laboratories will be coordinated as follows:

- A DOE Management Committee will be established, comprising Ian Foster (ANL: Chair), Chris Ding (LBNL/NERSC), John Drake (ORNL), Phil Jones (LANL), Jay Larson (ANL), and Doug Rotman (LLNL). This group is responsible for monitoring the progress of the project, reporting progress, and adapting plans (including resource allocations, if necessary) to ensure success.
- In addition, John Drake and Jay Larson will take on the roles of coordinators for DOE atmosphere model and coupler development, respectively. These individuals will be responsible for day-to-day coordination of these activities and liaison with NCAR.
- The DOE Management Committee will use biweekly telecons as their primary communication mechanism. In addition, regular meetings will be held for the entire collaboration (subject to travel constraints) as well as smaller meetings focused on individual components.
- We will prepare regular reports to DOE and NSF management on the progress of the project, noting milestones achieved and any problems encountered.

We hope that NCAR will agreed to appoint individuals to participate in this structure (e.g., a management committee co-chair, atmosphere model coordinator, coupler coordinator, and perhaps also CSM and PCM user representatives on the management committee), hence creating a joint DOE-NCAR management structure. It may also be appropriate to include NASA representatives.

The tasks proposed here are compatible with the scientific goals of both the CSM Scientific Steering Committee (SSC) and DOE's CCPP Program. As an important and crucial addition to the CSM working group structure, this DOE-NCAR-NASA collaboration will provide critical mass for the new Software Engineering Working Group. The goal of this working group is to provide visibility to the needed software issues associated with the CSM science capabilities and to provide an avenue of interaction with the other science-based working groups. We will look this group to provide scientific oversight for the work proposed here. In addition, we plan to incorporate selected DOE scientists as strong members in the CSM science working groups—namely the atmosphere, ocean, biogeochemistry, polar processes, and land system working groups.

The DOE Climate Change Prediction Meeting (CCPP) will act as the primary scientific meeting for this project. Activities outside the DOE/NCAR complex will also be involved. Most importantly here is the considerable expertise and ongoing NCAR collaboration with the NASA Data Assimilation Office (NASA DAO). Prime contact at the DAO will be Ricky Rood. The existing collaboration between the DAO and NCAR towards a next generation atmospheric model will benefit this DOE/NCAR effort considerably.

The activities and products of this activity that a joint DOE/NCAR/NASA team will develop will be captured in large part in design documents for the atmospheric component and the flux coupler that will guide the software and computational issues for an improved CSM. This document will include CSM scientific and DOE application requirements as well as a discussion of the needed software structure. Detailed analysis of the functional interfaces between major modules as well as within modules is essential before managed model development can be carried out. As these design documents are developed, the tasks, responsibilities, and timelines outlined in this proposal will be refined. Responsibilities for design and implementation of each model component will be assigned with timelines for the deliverables.

7 Relationship to Other Activities

The proposed R&D complements a variety of activities at NCAR, NASA, and elsewhere.

7.1 Data Management

The Earth System Grid project, a DOE-funded collaborative effort involving ANL, LANL, LBNL, LLNL, NCAR, and USC/ISI, has as its goal the prototyping of a system that will support the following:

- The rapid transport of climate data between centers and users in response to user requests. The focus is on end-user accessibility and ease of use, which will be accomplished through both the modification of existing applications and tools (e.g., PCMDI data analysis tools) and the development of new tools as needed to operate seamlessly in the Earth System Grid.
- Integrated middleware and network mechanisms that broker and manage high-speed, secure, and reliable access to data and other resources in a wide area system.
- A persistent Earth System Grid testbed that provides virtual proximity and demonstrates reliable high-performance data transport across a heterogeneous environment.

The technologies to be developed in this project contribute to the goals of the current activity by making it possible to manage the large datasets that will be produced by a high-performance CSM. The following table summarizes the data rates that the ESG project seeks to support.

Time frame	**1993-97**	**1999**	**2000-01**
Computer speed, peak	~ 100 GF	1 TF	5 TF
Production rate (MB/s)	~ 2	~ 20	~ 100
Intersite network	155	622	622
Peak rate (MB/s)	19	78	78
In practice (MB/s)	~ 0.3-3	~ 50	~ 50

7.2 Computing Technologies

The research proposed here is expected to benefit from a wide variety of other research activities on high-performance computing being performed within DOE and elsewhere. We mention just a few of the more important examples here.

Common Component Architecture Forum. The multilaboratory CCA Forum is investigating techniques for the modular construction of complex, high-performance scientific simulation codes. We are hopeful that techniques developed by this group will prove useful in the development of a more modular CSM.

High Performance Message Passing Interface. Researchers at Argonne are investigating techniques for efficient MPI execution on large parallel computers (e.g., efficient collective operations) and on systems supporting hybrid distributed/shared memory programming models. This is a critical issue for CSM execution.

8 Tasks and Milestones

The following table summarizes the tasks and milestones associated with this project. The table indicates activities undertaken by the DOE-NCAR-NASA team with deliverables and designations of responsibility. The institutional responsibility does not indicate that the activity is performed solely at that institution or that funding for the institution is commensurate with the task. But it is the responsibility of the institution to see that delivery is made using the resources of the collaboration. All the activities will involve coordination of multiple institutions and the delivery schedule is aggressive given this fact. Whenever possible, deliverables will be met ahead of schedule and we may swap some of the scheduled times of later tasks to compensate for unforeseen complications.

Activity	**Description**	**Deliverable (Responsility)**	**Schedule**

CCM preliminary design	A design of CCM defining goals and software interfaces	Document (NCAR)	6 months
CCM dycore interface	A modular CCM code that implements the designed interface	Preliminary modular CCM code (NCAR)	6 months
CCM-4 Utility and Math library	Library implementing data transposition, halo updates, and transforms for parallel decompositions	Tested code compatible with modular CCM (ORNL)	6 months
CCM Parallel Physics Design	Planning of the column physics package update to allow a specifiable run as the inner index, optimization for cache, optimization for vector.	Tested code and characterization of performance (NCAR)	6 months
CCM Parallel Physics Optimization	Implementation of the column physics package to allow a specifiable run as the inner index, optimization for cache, and optimization for vector.	Tested code and characterization of performance (ORNL)	12 months
CCM (Lin-Rood) dynamical core	A scalable, parallel implementation of the Lin-Rood scheme conforming to the CCM-4 dycore interface	Tested, documented Lin-Rood code (LLNL)	12 months
CCM (Eulerian Spectral) dynamical core	A scalable, parallel implementation of the reduced grid Eulerian Spectral scheme conforming to the CCM-4 dycore interface	Tested, documented Eulerian/Spectral code (ORNL)	12 months
CCM (Semi-Lagrangian) dynamical core	A scalable, parallel implementation of the reduced grid Semi-Lagrangian/Spectral scheme conforming to the CCM-4 dycore interface	Tested, documented Lagrangian/Spectral code (ORNL)	12 months
CCM integration	A scalable, parallel implementation of CCM-4 which includes the three dynamical cores.	Scalable CCM tested and available (NCAR)	12 months
Full CCM Software Engineering design	A full software engineering design exercise for the atmospheric component of CSM	Design document (NCAR)	12 months
POP (barotropic solver)	Performance improvement of the barotropic solver for coarse resolutions	Demonstrated performance in POP (LLNL)	6 months
Ice model	Performance study and improvement of ice model	Demonstrated improvement in CICE (LLNL)	12 months

Coupler performance study	Performance study of current coupler (PCM and CSM)	Performance report and documented ideas for improvement (ANL)	6 months
Coupler performance optimization	Performance optimizations to the coupler used within the current CSM	Performance demonstrated in CSM (ANL)	6 months
Preliminary Flux Coupler design	A preliminary design document to improve the scalability of the CSM flux coupler	Design document (NCAR)	6 months
CSM (coupler) Regridding Software	Parallel re-gridding software that effectively deals with load imbalances and scaling	Tested, documented coupler regridding code (LANL)	12 months
CSM (coupler) Object Framework	An object oriented framework for the flux coupler that solves the handshake problem.	Tested, documented coupler framework code (ANL)	12 months
Coupler extensions	Investigations of coupling tropospheric chemistry with CSM	Design document (ANL)	18 months
Source code repository	Develop centralized source code repository	Source code repository operational for CSM (NCAR)	6 months
Automated build/test infrastructure	Develop automated methods for building and testing CSM-2 components and entire model	Automated build-test operational (ANL)	12 months
Model performance characterization	Detailed performance characterization of current CSM and PCM on NERSC computers	Detailed report (LBNL)	6 months
CSM I/O Framework	Incorporates parallel I/O support for history and restarts of component models	Implementation on target platforms and performance characterization (LBNL)	18 months
Full CSM (coupler) Design	Software engineering design for CSM flux coupler	Design document (NCAR)	18 months
CSM integration	Fully integrated scalable CSM which incorporates CCM-4, POP, coupler	Tested, documented scalable CSM code (NCAR)	18 months

9 Resources

9.1 DOE Laboratories

The resources required to accomplish the proposed work are as summarized in the table.

Task	**Who**	**FY2000**		**FY2001**	
		$1,000	**FTEs**	**$1,000**	**FTEs**

Atmosphere Model					
Lin-Rood	LLNL	165	0.6	330	1.3
Spectral	ORNL	210	1.0	420	2.0
Coupler					
Coupling framework	ANL	240	1.2	480	2.4
Coupler activity	LANL	65	0.3	130	0.5
I/O	LBNL	90	0.4	180	0.8
Other					
Barotropic solver	LLNL	65	0.3	130	0.5
DSM/SP optimizations	LBNL	90	0.4	180	0.8
Coordination	ANL	75	0.4	150	0.8
TOTAL		**1000**	**4.5**	**2000**	**9.0**

Note that we also expect to leverage the following resources:
- 1.5 FTE funded at LLNL to work on scalable parallel Lin-Rood.
- 1.5 FTE funded at ORNL to work on PCM issues.
- Staff funded at LANL to work on POP and couplers.

9.2 NCAR

The work proposed here can only move forward effectively if additional human resources are bought to bear at NCAR to provide local support for this new multi-institutional collaboration.

The basic function to be filled relates to the coordination of all details relating to code development and testing between the players involved. Tasks will include managing the code repository; scheduling, coordinating, and advertising code check-ins; verifying that standards are met; verifying tests have been done; and verifying that all required documentation (with respect to code and testing) is submitted with check-in.

We believe that this task requires at least two new positions; one focused on CCM and the other on other aspects of CSM, in particular the coupler. For various reasons, these people should be located at NCAR and employed as part of the core NCAR staff.

A. The Scripps/NCAR/PNNL ACPI Pilot Project

This appendix provides a brief summary of a related DOE-funded project, the “ACPI Pilot Project” proposed by Scripps, NCAR, and PNNL (PI: Tim Barnett, Scripps Institute of Oceanography). The work proposed here is intended to support that project by providing enhanced performance global modeling capabilities.

a. Introduction

The National Research Council has recently acknowledged what has been known for the past five years: "the United States lags behind other countries in its ability to model long-term climate change" (NRC, 1999). It further finds "it is inappropriate for the United States to rely heavily upon foreign centers to provide high-end modeling capabilities".

The practical ramifications of these findings for the future energy policy of the United States are incalculable. They mean that when negotiating possible future energy policy on the international stage, such as in Kyoto, Buenos Aires, and so forth, the United States is at a distinct disadvantage. It simply does not have its own scientific base upon which to decide major issues. Rather, it must rely on simulations of future world conditions produced by other nations. Given the current state of the art, there is considerable room for interpretation in such simulations. In short, given the immense economic implications, the United States has no choice but to develop its own technical basis for making future energy policy decisions.

A bold solution to the unacceptable situation described above has been put forth in DOE's Accelerated Climate Prediction Initiative (ACPI). This large, long-term program would acquire and put in place the computational resources required for the future world simulations, while simultaneously developing the scientific and other infrastructure needed to carry through a full assessment of potential anthropogenic threats. This information would then allow the United States to develop a rational, quantitative basis for energy policy decisions.

b. Overview of Proposed Program

The basic idea is to begin immediately a pilot ACPI effort, using existing tools and previously demonstrated capabilities. This pilot will jump-start the full program, which will take several years to spin up. The pilot is intended to allow a partial scoping of the full ACPI to determine just what needs to be done and where the main difficulties are apt to occur. In order to do this, the pilot will take an 'end-to-end' approach, starting from the current state of the global climate system and ending up with quantitative statements about predicted anthropogenic impacts at the local and regional level. We expect these impact statements, while limited in number as befits a startup program, to be practically useful. Perhaps most important, the successful completion of the pilot will be a demonstration, or existence proof, that the ACPI is both feasible and likely endowed with major payoffs. The pilot will last two years and leave behind what will be the core of the full-scale ACPI.

In brief, the pilot ACPI is composed of three main elements. Element 1 uses existing ocean observations and inverse techniques to quantitatively establish the physical state of the global ocean in recent years. This information will serve as initial conditions for a coupled global climate model, which, when forced with anthropogenic pollution scenarios for the next century, will provide predictions of expected climate change globally. Recent work has shown ensembles of such runs are required to adequately identify the anthropogenic signals. The third program element downscales these large scale predictions of the global model to ensembles of regional-scale predictions, which are then used to estimate impacts on hydrology, agriculture, energy utilization, and so forth. We have already demonstrated the ability to do nearly all of the steps described above. In short, there is essentially no doubt we have the tools and technology to successfully carry through the pilot ACPI, if the necessary resources (especially computer capacity) are made available.

c. Main Program Elements and Their Status

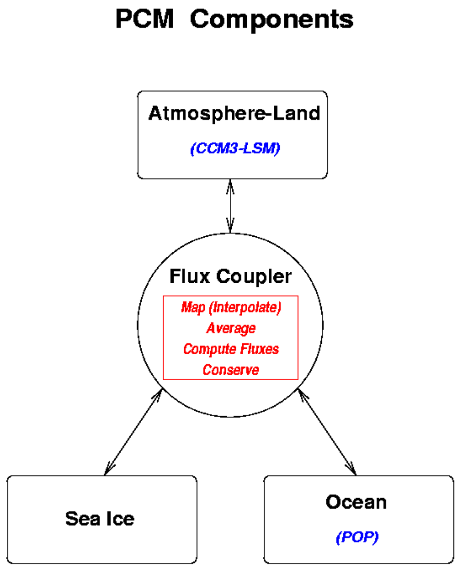
The current status of the three program elements is described below. All are currently operational.

i. Element 1: Ocean Assimilation

It has recently become possible to assimilate observed ocean data on a global scale, thereby producing gridded fields of such quantities as temperature, salinity, and velocity for the world oceans. etc.)

ii. Element 2: Modeling Anthropogenic Climate Change

The ocean data described above will be used to initialize the DOE-supported Parallel Climate Model (PCM), a coupled ocean/atmosphere/ice model being developed with mostly DOE support and collaborative NSF support. As the name indicates, the model is designed to work on highly parallel computers.



Schematic of the model components and interactions.

Details of the model may be found at http://www.cgd.ucar.edu/pcm. Suffice to say that the PCM has been developed with substantial distributed involvement of both government laboratories and universities. The NCAR/UCAR NSF-supported Climate System Model (CSM) project is working closely with the DOE community to develop new-generation climate models. Recently, it was agreed that they would use the same components at several different resolutions with implementation on a variety of supercomputers. Cooperation with the CSM will provide added academic involvement in this project.

The atmospheric component of the PCM is the CCM-3 atmospheric general circulation model (AGCM) developed at NCAR and is used at T42 resolution (about 280 by 280 km grid). The CCM-3 includes a land surface model that accounts for soil moisture, vegetation types, etc, as well as a river transport model. The University of Texas (Austin) parallel river transport model is a new component to the PCM. The ocean component of PCM is the Parallel Ocean Program (POP) model developed jointly by LANL, Naval Postgraduate School (NPS), and NCAR. We note that although it is not presently included in POP, a biogeochemical component could be added to make this system serve the needs of the national carbon cycle plan. As part of the effort to develop a newer generation of the PCM, the Gent-McWilliams parameterization of subgrid-scale eddy contributions along isopycnal layers has been added along with the K-profile parameterization mixing in the upper ocean. The final major model component of PCM is a sea ice model developed at NPS. The sea ice model in the first version of PCM is that used by Y. Zhang at NPS. In the second version of the PCM, the sea ice model uses the University of Washington multi-thickness thermodynamical model of C. Bitz and the elastic-viscous-plastic ice dynamics model by E. Hunke of LANL. These new features will make the sea ice treatment more realistic. The sea ice formulation is especially important for reproducing realistic feedback mechanisms between sea ice processes and climate change forcing.

The full PCM has been configured to execute with a serial flux coupler that has been designed to perform the calculation of the components of the climate system as efficiently as possible on a variety of parallel high-capacity supercomputers. Specifically, the PCM can run on the IBM SP, Origin SGI, Cray T3E, and Compaq and Linux Beowulf systems. The present version makes use of the message-passing interface (MPI) for passing information between processors and nodes. PCM version 2 will have a capability of using a hybrid approach in which MPI is used between nodes and open message passing within a node. This will allow efficient use on virtually all the cluster parallel computer systems that are being developed as supercomputers.

The PCM is currently fully operational. Analyses of ongoing simulations have shown realistic amplitude El Niño, La Niña, North Atlantic Oscillation, and Antarctic Circumpolar Wave properties in the simulations (see, for example, Washington et al. 2000; Semtner 2000; and Meehl et al. 2000). Weatherly and Zhang (2000) have examined the polar aspects of the PCM. The PCM effort has already produced seven ensemble anthropogenic scenario predictions. The ensemble simulations start from year 1870 and simulate the climate to the year 2000 with observed greenhouse gas concentrations and sulfate aerosol concentrations. Then for the predictions from year 2000 to 2100, the forcing scenarios are used. The forcing scenarios are the same as those used in the NCAR Climate System Model studies of climate change. The greenhouse gas and sulfate aerosol concentrations were obtained from an earlier CSM simulation. By way of example, Figure X shows the change in near surface air temperature projected to occur when future levels of CO_2 concentration reach twice their current values from an ensemble of 1% CO_2 per year increase simulations (courtesy of W. Washington). It is information from large ensembles of runs such as this that will be used to drive the regional climate modeling effort and provide ensemble statistics with which to judge the simulations.

To better facilitate use of the PCM model data for regional climate change studies, a doubling of the number of gridpoints in each horizontal direction will be made in both the ocean and atmosphere models as the two-year project progresses. These higher-resolution component models already exist and are being tested. The present model is a T42 atmospheric CCM model with an approximately 2.5 degree horizontal resolution. A T85 version of CCM is being tested and some simulations with a resolution twice the present will be carried out over the two-year period. When the higher-resolution atmospheric model is coupled into the PCM, there will be better output for the downscaling efforts described below. However, for the first stages of the project, we will use the existing T42 resolution output from the ensemble of simulations. A list of CCM variables that can be used for analysis and boundary conditions for a regional climate model can be found at http://www.cgd.ucar.edu/pcm/PCMDI.

iii. Element 3: Regional Downscaling/Impacts

A surprisingly broad capability exists to take the large-scale predictions made by global climate models and use them to force regional-scale atmospheric models. The resulting high-resolution estimates of physical climate variables (temperature, precipitation, snow, etc.) are subsequently used, for example, in hydrological and agriculture models to provide estimates of economic impacts of climate change.

The pilot effort will concentrate on anthropogenic impacts on the hydrological system of the western United States. At least three different downscaling techniques and subsequent applications to water issues will be used. Two of the methods involve dynamical models driven by the atmospheric component of PCM, the CCM-3. Both of these dynamic models are currently operational using input from CCM-3 to make regional climate forecasts at seasonal-to-interannual time scales and of anthropogenic impacts, just what we want to do in the pilot. These models need to be run in the ensemble mode for our purposes.

Bibliographic References

Ashworth, M. 1999. Optimisation for Vector and RISC Processors," in *Towards Teracomputing*, World Scientific, River Edge, NJ. pp. 353-359.

Bitz, C.M., Holland, M.M., Eby, M. and Weaver, A.J. 2000. Simulating the ice thickness distribution in a coupled climate model. *J. Geophys. Res*., submitted.

Bitz, C.M. and Lipscomb, W.H. 1999. An energy-conserving thermodynamic model of sea ice. *J. Geophys. Res.* **104**, 15669-15677.

Bonan 1998. The land surface climatology of the NCAR land surface model coupled to the NCAR community climate model, *J. Climate*, **11**, 1307-1326.

Briegleb and Bromwich 1998. Polar radiation budgets of the NCAR CCM-3, *J. Climate*, **11**, 1246-1269.

Brooks, Frederick P., Jr., The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition, Reading, Massachusetts, Addison-Wesley 322 pp, 1995.

Armstrong, R., Gannon, D., Geist, A., Keahey, K., Kohn, S., McInnes, L., Parker, S., and Smolinski, B. 1999. Toward a comon component architecture for high-performance scientific computing, to appear in *Proceedings of the 1999 High Performance Distributed Computing Conference*.

C. M. Bitz, M. M. Holland, M. Eby and A. J. Weaver. 2000. Simulating the ice-thickness distribution in a coupled climate model. *J. Geophys. Res*., Submitted.

Climate System Model 1998. Special issue, *J. Climate* **11,** no. 6.

Colella, P., and Woodward, P. 1984. The piecewise parabolic method (PPM) for gas-dynamical simulations, *J. Comput. Phys*., **54**, 174-201.

DAO 2000: Algorithm Theoretical Basis Document, Version 2.0, Data Assimilation Office, NASA Goddard Space Flight Center, Greenbelt, Maryland 20771. Available online at http://dao.gsfc.nasa.gov/subpages/atbd.html.

Ding, C. H. Q., and Y. He. 1999. Data organization and I/O in a parallel ocean circulation model, *Proc. SC99,* November.

Drake, J., Foster, I., Michalakes, J., Toonen, B., and Worley, P. 1995. Design and performance of a scalable parallel community climate model, *Parallel Computing*, **21**, 1571-1592.

Dukowicz, J. K., and R. D. Smith. 1994. Implicit free-surface method for the Bryan-Cox-Semtner ocean model, *J. Geophys. Res.* **99,** 7991-8014.

Dukowicz, J. K., R. D. Smith, and R. C. Malone. 1993. A reformulation and implmentation of the Bryan-Cox-Semter ocean model on the Connection Machine, *Atmos. Ocean Tech*. **10,** 195-208.

Flato, G., and Hibler, W. 1992. Modelling pack ice as a cavitating fluid, *J. Phys. Oceanogr*., **22**, 636-651.

Gates, W.L. et al. 1996, Climate Models - Evaluation, Chapter 5 of the Intergovernmental Panel on Climate Change Second Scientific Assessment of Climate Change, CUP, pp. 233-276.

Gent, P. R., and J. C. McWilliams. 1990. Isopycnal mixing in ocean circulation models, *J. Phys. Oceanography* **20,** 150-155.

Hack, J. 1994. Parameterization of moist convection in the NCAR community climate model, *J. Geophys. Res*., **99**, 5541-5568.

Hack, J., Kiehl, J., and Hurrell, J. 1998. The hydrologic and thermodynamic characteristics of the NCAR CCM-3, *J. Climate*, **11**, 1179-1206.

Holtslag, A., and Boville, B. 1993. Local versus nonlocal boundary-layer diffusion in a global climate model, *J. Climate*, **6**, 1825-1842.

Hunke, E. and Dukowicz, J. 1997. An elastic-viscous-plastic model for sea ice dynamics, J. *Phys. Oceanogr*., **27**, 1849-1867.

Hunke, E. C., and W. H. Lipscomb. 1999. CICE: The Los Alamos Sea Ice Model, Documentation and Software, Version 2, Los Alamos National Laboratory, LA-CC-98-16 v.2,

Hurrell, J., Hack, J., Boville, B. 1998. The dynamical simulation of the NCAR community climate model CCM-3, *J. Climate*, **11**, 1207-1236.

Jones, P. W. 1999. First- and second-order conservative remapping schemes for grids in spherical coordinates, *Monthly Weather Rev*. **127,** 2204-2210.

Kattenberg, A., F. Giorgi, H. Grassl, G.A. Meehl, J.F.B. Mitchell, R.J. Stouffer, T. Tokioka, A.J. Weaver, and T.M.L. Wigley, 1996: Climate Models - Projections of Future Climate. In Climate Change 1995 the Science of Climate Change The Second Assessment Report of the IPCC: Contribution of Working Group I (Eds. Houghton, J.T., L.G. Meira Filho, B.A. Callander, N. Harris, A. Kattenberg, and A. Maskell), Cambridge University Press, 285-357.

Kiehl, J., Hack, J., Bonan, G., Boville, B., Williamson, D., and Rasch, P. 1998. The National Center for Atmospheric Research community climate model: CCM-3, *J. Climate*, **11**, 1131-1149.

Kiehl, J., Hack, J., and Hurrell, J. 1998. The energy budget of the NCAR community climate model CCM-3, *J. Climate*, **11**, 1151-1178.

Large, W. G., J. C. McWilliams, and S. C. Doney. 1994. Oceanic vertical mixing: A review and a model with a non-local boundary layer parameterization, *Rev. Geophysics* **32,** 363-403.

Lin, S-J. 1997. A finite-volume integration method for computing pressure gradient force in general vertical coordinates, *Quart. J. Roy. Meteor. Soc.* **123**, 1749-1762.

Lin, S-J and Rood, R.B. 1996. Multidimensional flux-form semi-Lagrangian transport schemes, *Mon. Wea. Rev*. **124**, 2046-2070.

Lin, S-J and Rood, R.B. 1997. An explicit flux-form semi-Lagrangian shallow water model on the sphere, *Quart. J. Roy. Meteor. Soc.* **123**, 2477-2498.

Lin, S.-J., and Rood, R.B. 1998. A flux-form semi-Lagrangian general circulation model with a Lagrangian control-volume vertical coordinate. The Rossby-100 symposium, Stockholm, Sweden.

Lin, S.-J., and Rood., R.B. 1999. Development of the joint NASA/NCAR General Circulation Mode. Preprint, 13th conference on Numerical Weather Prediction, Denver, CO.

McConnell, S. 1993. *Code Complete, a Practical Handbook of Softwware Construction*, Redmond, Washington: Microsoft Press.

Meehl, G.A., Gent, P., Arblaster, J., Otto-Bliesner, B., E. Brady, E., and Craig, A. 2000. Factors that affect amplitude of El Niño in global coupled climate models. In preparation for Climate Dynamics.

Michalakes, J. 2000. The same source parallel MM5, *Scientific Programming* (accepted).

Michalakes, G., Dudhia, J., Gill, D., Klemp, J., and Skamarock, W., 1998. Design of a next-generation weather research and forecast model,' in *Proceedings of the Eighth Workshop on the Use of Parallel Processors in Meteorology*, European Center for Medium Range Weather Forecasting, Reading, U.K., November 16-20, 1998. Available as ANL/MCS preprint number ANL/MCS-P735-1198.

Rosinski, J.M. and D.L. Williamson, 1997: The accumulation of the rounding errors and port validation for global atmospheric models. *SIAM J. Sci. Comput*. **18**, 552-564.

Semtner A., 2000. Ocean and climate modeling on advanced parallel computers: progress and prospects, *Communications of the ACM* (in press).

Smith, R. D., J. K. Dukowicz, and R. C. Malone. 1992. Parallel ocean general circulation modeling, *Physica* **D 60,** 38-61.

Smith, R. D., S Kortas, and B. Meltz 1995. Curvilinear coordinates for global ocean models, Los Alamos National Laboratory Report LAUR-95-1146.

Sterling, T., P. Messina, and P. H. Smith 1995. *Enabling Technologies for PetaFLOPS Computing,* The MIT Press, Cambridge.

Van Leer, B. 1977. Toward the ultimate conservative difference scheme. Part IV: A new approach to numerical convection. *J. Comput Phys*., **23**, 276-299.

Washington , W.M., 1982: Documentation for the Community Climate Model (CCM) version 0. NCAR report, Boulder, Colorado, NTIS No. PB82 194192

Washington et al. 2000. Parallel climate model control and transient simulations. *Climate Dynamics*, in press.

Weatherly, J.W. and Y. Zhang, 2000. The response of the polar climate to increasing CO2 in a global climate model with elastic-viscous-plastic sea ice, Accepted to *Journal of Climate*.

Winton, M. 1998. A reformulated three-layer sea ice model, Submitted to *Journal of Atmospheric and Oceanic Technology*.

Williamson, D.L. L.M. Bath, R.K. Sato, T.A. Mayer, and M.L. Kuhn 1983: Documentation of NCAR CCM0B Program Modules. NCAR Technical Note NCAR/TN-212+IA, Boulder, Colorado, NTIS No. PB83 263996

Williamson and Rosinski, 2000: Accuracy of reduced grid calculations. QJRMS, in press

Zhang, G., and McFarlane, N. 1995. Sensitivity of climate simulations to the parameterization of cumulus convectionin the Canadian Climate Centre general circulation model, *Atmos-Ocean*, **33** 407-446.